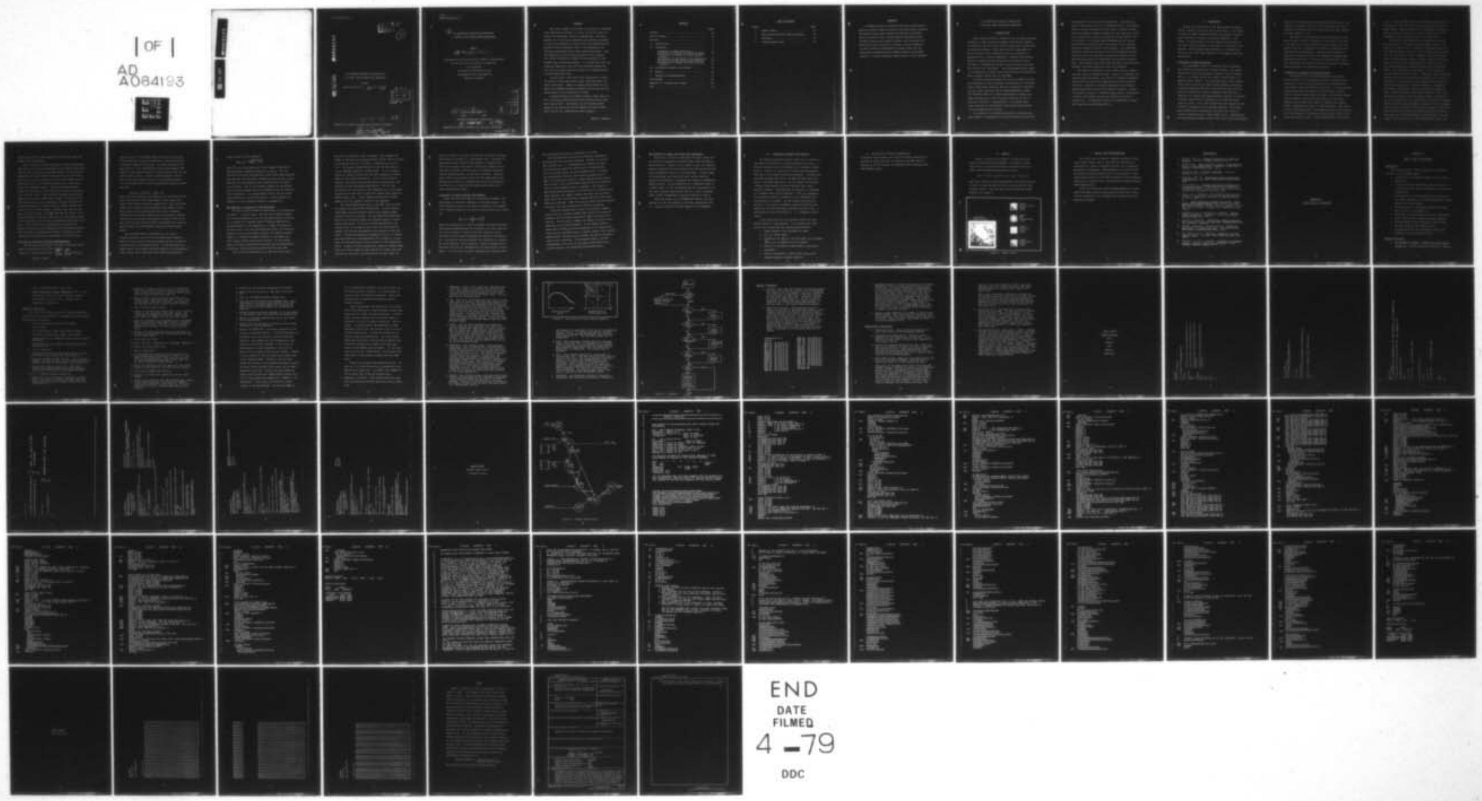
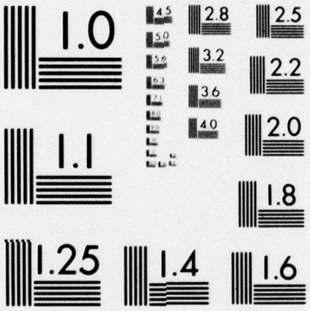
AD-A064 193 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 9/4
AN INTERACTIVE ANALYSIS PROGRAM FOR A DIGITAL IMAGE PROCESSING --ETC(U)
SEP 78 D A MCGAUGH
UNCLASSIFIED AFIT/GE/EE/78S-16 NL

| OF |
AD
A064193



END
DATE
FILMED
4 -79
DDC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AFIT/GE/EE/78S-16

AN INTERACTIVE ANALYSIS PROGRAM FOR A DIGITAL IMAGE PROCESSING LABORATORY

THESIS

AFIT/GE/EE/78S-16

Dennis A. McGaugh
Capt USA

Approved for public release; distribution unlimited.

JOSEPH P. HIPPS, Major, USAF
Director of Information
19 Jan 79

14

AFIT/GE/EE/78S-16

6

AN INTERACTIVE ANALYSIS PROGRAM FOR A DIGITAL IMAGE PROCESSING LABORATORY

THESIS

9 Master's thesis,

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University (ATC)
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by
10 Dennis A. McGaugh, B.S.
Capt. USA
Graduate Electrical Engineering

11 Sep[redacted] [redacted]78 12 74p.

Approved for public release; distribution unlimited.

012225

Preface

This work attempts to provide the solution to a problem first identified by Major J. W. Carl of the Air Force Institute of Technology, School of Engineering. The basic problem was to design and implement the software necessary to allow two-dimensional linear filter and two-dimensional Discrete Fourier Transform (DFT) to be performed on an existing minicomputer-based image processing system. With the addition of the routines described in this thesis, true linear filtering of images is now available on the DICOMED Image Processing System. It is hoped that with its filtering capability, and with its 64x64 DFT capability that this new image analysis software will indeed provide a useful analytical tool.

I would like to take this brief opportunity to thank the people without whom this project would have been virtually impossible. Thanks to Captain Fred Barney for his making the DICOMED equipment available and for his patient explanations on the intracacier of same.

A special thanks to my thesis advisor, Major Carl for his saint-like patience and profound guidance during some very trying times. And lastly, but by no means least, thanks to my wife Marla for her typing the thesis rough draft and for her inspirational moral support.

Dennis A. McGaugh

Contents

List of Figures

Abstract

A FORTRAN program is presented that will interactively prepare digitized images (pictures) for experimental analysis. The digital preparation includes the ability to Discrete Fourier Transform (DFT), Inverse Fourier Transform (IDFT), threshold image power spectral density, and to perform two-dimensional image filtering. Further, a brief discussion of image distortion measurements, optimal image size for experimental presentation, and image aliasing is provided. A sample of typical processed image results is also supplied.

AN INTERACTIVE ANALYSIS PROGRAM FOR A DIGITAL IMAGE PROCESSING LABORATORY

I. Introduction

There currently exists, located in the Air Force Avionics Laboratory, image processing hardware and software capable of digitizing and performing limited statistical calculations of pictorial imagery. There also exists a requirement to perform certain image processing with this equipment in preparation for a visual experiment. However, the software needed to conduct the processing did not formerly exist. The types of computations needed to perform this processing and analysis are well-known. But these computations needed to be developed in software and implemented on the particular small computer system that is available.

The image processing functions that are needed include: two-dimensional linear filtering, calculation of a two-dimensional image power spectrum, ability to specify a threshold for this power spectrum, and the Mean-Square-Error (MSE) and Bandwidth (BW) of an image constrained to occupy only those frequencies at which the power spectrum is above a specified threshold. Additionally, it is desired that these functions be interactively controlled and specified to the program during run time.

The problem is to develop and provide a software program capable of accomplishing the above functions, and still be compatible with the existing equipment. Additionally, part of this thesis work is to determine what is the minimum size of picture digitization (e.g. 64x64, 128x128, 256x256) that will produce pictures suitable for the experiment; that is, a picture that creates an image of no less quality than 30 lines per 1° of visual angle subtended during viewing. The picture digitization will also impact on the computer resources needed to perform the filtering. The experiment currently calls for the use of up to 24 different pictures. These pictures are available in positive or negative form, and have been digitized (2048x2048) and stored on magnetic tape. The sampling resolution may also cause image aliasing problems, and this effect must also be discussed.

The material presented in this thesis follows roughly the same order as discussed above. Chapter II discusses the constraining environment, its effect on program development, and the digital domain for implementation of the required functions. Chapter III briefly describes the laboratory hardware/software and the final project software. Chapter IV presents a sample of processed image results, and Chapter V closes with summary remarks and mention of areas for future work. The appendix contains a program user's guide and complete program documentation.

II. Discussion

During the development of the image analysis program, several important questions had to be answered. The answers to some are rather straight-forward; the answers to others were influenced by answers to previous questions. The answer to each question, and its impact on the design of the image analysis program is presented in the section that follows. After reading these sections, it is hoped that there will be clear understanding of the program constraint environment.

A Question of Image Resolution

The requirement exists to conduct visual experiments with pictorial presentation to human subjects. The images should fall within the subjects' regions of foveal vision; that is, the images should not exceed a visual angle of 2 degrees during viewing. In discussions with (then) Captain Larry Goble of the US Air Force Flight Dynamics Laboratory at Wright-Patterson AFB, Ohio, the point was made that the industry standard for images on graphics and television displays is a twelve-inch square picture viewed at a distance of thirty inches with a resolution of no less than sixty lines per inch. It was also decided that this should be the resolution standard for the analysis program. After some basic geometrical calculations, this works out to be 30 lines of resolution per 1 visual degree (vis). Additionally, the experimental displays require that each visual presentation consist of two different pictures placed side-by-side. Considering the overall limit of 2° vis vertically and horizontally, this means that for square pictures, each picture can only subtend 1° vis.

The answer to the question of image resolution is that each picture, in order to subtend 1° vis, must be 1/2 inch x 1/2 inch at 30 inches distance from the viewer, and each picture must have a minimum of 30 lines of resolution both horizontally and vertically. Imagery of this quality is obtainable through the use of the laboratory DICOMED image processing equipment. It should be noted, however, that the requirement for number of lines of resolution increases when considering the question of aliasing.

A Question cf Aliasing in Digitized Images

The DICOMED image processing equipment digitizes a given picture with a sequential horizontal scanning action. It begins at the top of the picture to be digitized with a horizontal scan the width of the picture. It then proceeds incrementally down the picture with successive, picture width, horizontal scans until it stops at the picture bottom. Both horizontal and vertical directions are discretely sampled with an adjustable number of samples per picture widths (independently chosen in each direction). The result of this scanning process is the mapping of the entire picture into a MxN matrix of M·N discrete pixel values, where M and N are both powers of two. In this application, M was equal to N. For the DICOMED machine, the value of each pixel can be any one of 256 gray scale levels, from white (377_8) to black (000_8). When referring to the resolution of this resulting digitized picture, it is said to have N lines of resolution, with a sampling rate of N (where $N=2^P$).

The answer to the question of aliasing can be found by referring to the Nyquist sampling theorm (12:68). Restated, it simply says: to eliminate spectrum aliasing in the discrete domain, sample at a rate greater than or equal to twice the spectral width. Therefore, for the previous case of an NxN sampled image, in order to prevent aliasing after discrete Fourier transformations, there must not be any spatial-frequency content above $\frac{N}{2}$ cycles per picture width. The experimental requirement is for a minimum of 30 lines resolution per picture. This means that a digitized picture with this resolution can display up to 30 full oscillations (cycles) between white and black per picture in either the horizontal or vertical direction. Therefore, the display resolution criterion is satisfied if $\frac{N}{2} \geq 30$. Since a DFT/FFT computer algorithim (discussed later) will be used to transform into the Fourier domain, and powers of two run fastest in that algorithim, and since the DICOMED image processing system samples at powers of two, a sampling rate of 64 was selected as the minimum convenient value that N can have and still meet the display resolution criteria. This requires a 64x64 digitization of each image to be processed, and insures that, at this digitization level, the resolution limits of human observers will be met under the cited viewing conditions.

Now, to insure no image aliasing, it must be guaranteed that there is no spatial frequency, in the image being transformed, above 32 cycles per picture width. To provide this guarantee, all 64x64 digital images used in this work were derived from pictures sampled at a rate much higher than 64 samples per line. The original pictures were first digitized into 2048x2048 images. Then by considering 64x64 blocks of 32x32 pixels, each 32x32 pixel block was averaged into one pixel value. The resultant averaged pixel values produce the required 64x64 digital image. In essence, this procedure performs a two-dimensional, digital low-pass filtering operation on a picture that may contain spatial frequencies of up to 1024 cycles. Although this type of low-pass filtering results in a $\frac{\sin x}{x}$ form for the filter in the frequency domain, and such filters are not particularly good in the stop-band of the filter (considerable side-lobe structure), these effects were ignored since relatively low spatial frequencies are anticipated in the original pictures. This procedure is essentially a digital simulation of the low-pass filtering operation necessary to avoid aliasing.

The Question of Linear Filtering Implementation

The functional requirement for linear filtering implies spatial convolution of the form (10:440):

$$y(n_1,n_2) = x(n_1,n_2) * h(n_1,n_2) = \sum_{m_1=-\infty}^{\infty} \sum_{m_2=-\infty}^{\infty} x(m_1,m_2) \cdot h(n_1-m_1,\ n_2-m_2)$$

where $y(n_1,n_2)$ is the output image result of the $x(n_1,n_2)$ input image that has been filtered by a linear filter with an impulse response of $h(n_1,n_2)$. As noted by Rabiner and Gold (Ref 10), this calculation in the spatial domain with images is not usually soluable in closed form except for the most trivial filters. However, a more tractable form of this convolution equation occurs in the frequency domain. It is well-known (Ref 8) that the convolution equation takes the form:

$$Y(k_1,k_2) = X(k_1,k_2) \cdot H(k_1, k_2)$$

where $Y(k_1,k_2)$, $X(k_1,k_2)$, and $H(k_1,k_2)$ are the Fourier transform coefficients of the image output, image input, and image filter impulse response respectively. Now, the filter computations become readily manageable and almost trivial. They simply involve multiplying each Fourier transform coefficient of the input image with each respective Fourier transform coefficient of the impulse response of the filter. The result is then inverse Fourier transformed back into the spacial domain as the resultant linearly filtered output image.

The transfer function furnished for the linear filtering process is given as a one dimensional equation. Before it can be used to filter an image, however, it must be converted into two-dimensional form. It has been shown by Huang (Ref 6) that a good circularly symmetric two-dimensional linear filter can be obtained from a good one-dimensional linear filter via the relation:

$$H(k_1,k_2) = \hat{H}(\sqrt{k_1{}^2 + k_2{}^2})$$

where $\hat{H}$ is the one-dimensional filter impulse response at the appropriate frequency values of k_1 and k_2. For the situation involving the previously cited 64x64 image matrices, this requires Fourier domain frequency values (k_1,k_2), along both the real and imaginary axis, ranging from -32 cycles to +32 cycles. Due to the circular symmetry of the filter, only one 32x32 quadrant need be calculated from the equation. Once calculated, it is appropriately copied into the other three quadrants of the filter matrix. A more complete explanation of this operation appears in Appendix A.

The Question of Operating in Two Dimensions

The most obvious problem of processing in two-dimensions is simple: it takes longer, and uses more computer memory. For the one-dimensional case, the discrete Fourier transform of a sequence of N samples takes N^2 complex addition and multiplications; in two-dimensions, it takes N^4 operations by brute force. For the one-dimensional case, the filtering process in the Fourier domain requires $2 \bullet N$ complex coefficients to be stored in computer memory; in two-dimensions $2 \bullet N^2$ complex coefficients must be stored. To help reduce these increased requirements associated with two-dimensional processing, certain processing modifications were made.

First was the implementation of the well-known (Ref 8)

Fast Fourier Transform (FFT) technique. This reduces the number of addition and multiplications from N^4 (64^4=16,77,216) to $2N \cdot N\log_2 N$ ($128 \cdot 64 \cdot 6 = 49,152$), a reduction of over 10 to 1. The program implementation of the FFT algorithim is via the FORTRAN subroutine called FOURT. It is a program written by Norman Brenner from the basic program by Charles Rader of MIT Lincoln Laboratory. It is a very versatile program, and offers more options than are required for the particular image processing application. But its very versatility was the reason it was selected, as it will allow calculation of odd sizes of FFT's for images other than the present 64x64 size. A more complete discussion of FOURT appears with the program listing in the appendix.

The second modification was to use disk files to store the images and complex FFT coefficients. In the existing IPS/DICOMED image processing systems each complex number occupies 4 words of storage. For one 64x64 FFT coefficient matrix to be entirely loaded into computer memory it would require 16K words. If, as in a filtering operation, two matrices were to be used at once, then 32K words would be needed; the existing minicomputer core storage is only 28K words. A technique suggested by Oppenhein (Ref 8) to reduce the requirement for two-dimensional core storage is to perform the two-dimensional FFT as a series of one-dimensional FFT's. Further, by first storing the images on random access disk files then reading in only one row or column from each appropriate disk file, the FFT operates on only those two pieces of data at one time, and the core required to operate the program is dropped to a realizeable size. Although the processing scheme involves huge amounts of computer I/O, the time to compute and store a 64x64 point complex FFT is less than 4 minutes, and the time to accomplish filtering (matrix multiplication) is less than 2 minutes. Additionally, there is an option to have the FFT coefficients normalized by dividing all coefficients by the zero-frequency or DC term, or to have the FFT coefficients remain with their unnormalized values.

A Question of Power Spectrum Thresholding

Once the digitized image is in the Fourier domain, the calculation of the power spectrum is straight-forward. The power spectral density of an image is related to the squares of the absolute value of the Fourier coefficients and is given by:

$$\Phi(u,v) = E\left\{\left|X(u,v)\right|^2\right\}$$

where the expectation is over an ensemble of image transforms. The program implementation for estimating the power spectrum of a set of images is obvious from this equation, and a discussion can be found in Appendix A, Operating Procedures, 5c. The analysis program provides the capability to average up to 24 different image spectrums. These 24 images may be the same ones that are mentioned in the introduction to this paper. It is on this resultant averaged power spectrum that the thresholding function is expected to be used.

The thresholding function was originally requested to be used with only the power spectrum. However, to provide more flexibility, it is implemented in a manner to allow its use with any specified 64x64 matrix of complex data (i.e. FFT coefficients, or power coefficients). By using the thresholding function, the analysis program user is able to interactively specify a 64x64 complex valued matrix, and to enter a real valued number against which the absolute value of every complex point is compared. The result of this thresholding operation is the creation of a 64x64 matrix which contains (in complex format) the value one for every point above the threshold and a complex zero for every point below the threshold. Both the original matrix and the threshold matrix are then available for further processing.

Since the thresholding is done in the Fourier domain, values that are linearly related (Ref 1:138-141) to spatial bandwidth (BW) and mean-square-error (MSE) are easily calculated. The BW value is simply the number of points above threshold in the Fourier domain. In the spatial domain, these above-threshold values correspond to the image pixels that would be transmitted over a digital channel with relative bandwidth BW. The MSE value is simply the sum of the power values below threshold. In the spatial domain, the MSE corresponds to the error of the image transmitted over a digital channel, with bandwidth BW described above.

The Question of Images and Image File Generation

Originally, the only requirement for image output was to generate a completely processed image ready for experimental testing. However, in an effort to make the analysis program applicable for more general use, the ability to display intermediate results is also provided. A 64x64 image file displaying the results for each of the following functional operations is prepared and stored in DICOMED/IPS format on disk: (1) the reduction of a 2048x2048 image to a 64x64 image; (2) the two-dimensional FFT calculation; (3) the power spectrum calculation; and (4) a fully processed image (either thresholded, linearly filtered, or FFT'd).

Since the images are in DICOMED/IPS format, post-processing with the IPS system is trivially implemented (Ref 7). A sample image file format appears in the appendix.

III. Laboratory Hardware and Software

The imagery equipment used for this work is located in the Avionics Laboratory on Wright-Patterson AFB, Ohio. A detailed description of the equipment is included in Reference 7. Briefly, it consists of a PDP-11/45 minicomputer with 28K of core memory and many peripherals. Included are: one RPØ3 diskpack disk drive, two RKØ5 diskcartridge disk drives, one MTU1Ø tape drive, one 132 column line printer, a Conrac video monitor, and one Tektronix 4000 series graphics terminal. Additionally, the system is interfaced to a DICOMED image scanner/recorder. With this equipment, the capability exists to perform up to a 2048x2048 point digitization of pictures. As the DICOMED digitizes the picture, it is stored on magnetic tape for follow-on processing by the PDP-11/45 Image Processing System (IPS). A description of the IPS software is also included in Reference 7.

A full description of the software written for this thesis appears in the appendix. It is written for interactive use and provides the following capabilities:

1. Image reduction from 2048x2048 to 64x64.
2. FFT of a 64x64 image.
3. Calculation of $\Phi_i(u,v)$ for one image, or an average $\hat{\Phi}(u,v)$ of a standard set of 24 images.
4. Ability to threshold a 64x64 image as previously discussed.
5. Ability to generate a 64x64 filter from a user supplied impulse response equation.

6. Inverse FFT of Fourier coefficients.

By running this program, any one of the above processes can produce an image file that is stored on the RPØ3 disk for display by the IPS system, or hardcopy picture generation by the DICOMED system.

IV. Results

Figure 1 presents some samples of output from the digital image preparation program. It should be noted that Figure 1 was generated with an arbitrary high-pass filter whose transfer function has the form:

$$H(f) = 0.5(1 + \ln(f/10 + 0.1), \text{ for } 0 \leq f \leq 32$$

Also note, that the slight graininess of the 64x64 pictures is due to a blow-up factor of two (4 new points for each original point). This was done solely to provide larger pictures for Figure 1.

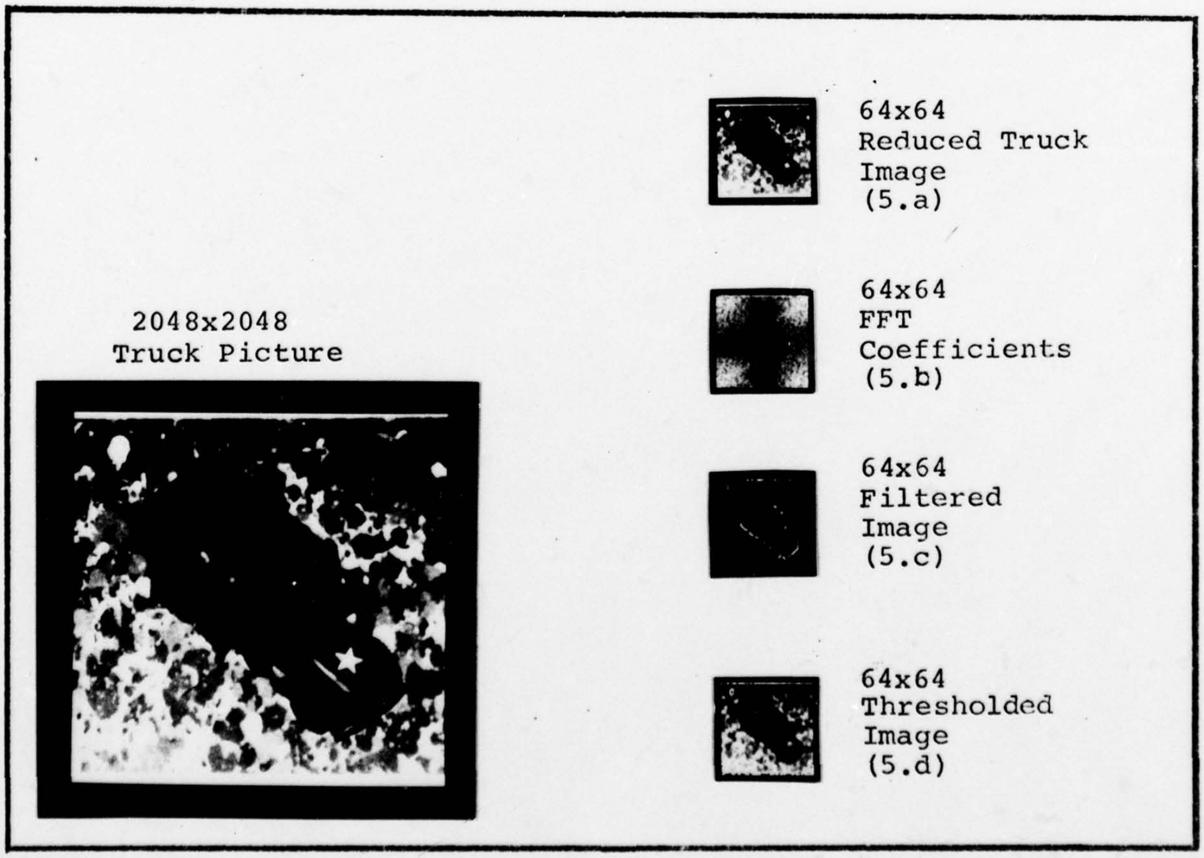


Figure 1. Sample Images

V. Summary and Recommendations

This thesis has presented a computer program to digitally process images, and in this case the processing is motivated by a desire to generate stimuli for a particular psychometric experiment. Additionally, as can be verified by reading the program description in the appendix, a general purpose routine is now made available to tailor digital processing of pictures for a large number of image filtering based applications.

Areas for future work would include modifying the program to process pictures other than the 64x64 size, and an on-line interactive tie-in between the IPS system and the processing program.

Bibliography

1. Andrews, Harry C. Computer Techniques in Image Processing. New York: Academic Press, 1970.

2. Berger, Toby. Rate Distortion Theory: A Mathematical Basis for Data Compression. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1971.

3. Cornsweat, Tom N. Visual Perception. New York: Academic Press, 1970.

4. Gallager, Robert G. Information Theory and Reliable Communication. New York: John Wiley and Sons, Inc., 1968.

5. Hall, Charles F. Digital Color Image Compression in a Perceptual Space. USCIPI Report 790. Image Processing Institute: University of Southern California, Los Angeles, California, February 1978.

6. Huang, T. S. "Stability of Two-Dimensional Recussive Filters," IEEE Trans. on Audio and Electroacustics, 2:158-163 (1972).

7. -----. Image Processing Software Conversion: Users Manual and Technical Report, Vol. I & II. Rome Air Development Center, Air Force Systems Command, Griffiss AFB, New York 13441. February 1977 (ADA038136 & ADA 038137).

8. Oppenheim, Alan V. and Ronald W. Schaffer. Digital Signal Processing. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975.

9. Papoulis, Athanasios. Probability, Random Variables, and Stochastic Processes. New York: McGraw-Hill, 1965.

10. Rabiner, Lawrence R. and Bernard Gold. Theory and Application of Signal Processing. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975.

11. Van Trees, Harry L. Detection, Estimation, and Modulation Theory. New York: John Wiley and Sons, Inc., 1968.

12. Ziemer, R. E. & W. H. Tranter. Principles of Communications: Systems, Modulation and Noise. Boston: Houghton Mifflin Company, 1976.

Appendix A

User's Guide for RAZMA.TAZ

Appendix A

User's Guide for RAZMA.TAZ

Introduction

This program allows a user to perform the following functions on IPS/DICOMED images:

1. A reduction of a 2048x2048 image file to a 64x64 image file.
2. A two-dimensional fast-fourier transform (FFT) of a 64x64 image with either normalized or unnormalized coefficients.
3. Calculation of a two-dimensional power spectrum of one 64x64 image file, or the two-dimensional average power of 24, 64x64 images.
4. Specification of a high-pass threshold on either the FFT file or the power file.
5. Generation of a two-dimensional filter (FFT domain) from a user specified impulse response equation.
6. Multiplication of any threshold or filter files with any image or FFT or power files.
7. Calculation of the two-dimensional inverse FFT transform on any modified FFT file.

Reference Material

1. The DOS/BATCH Handbook: Monitor Operating System Version Ø9. Digital Equipment Corporation, Maynard,

Mass. DEC-11-ODBHA-A-D, April 1974.

2. Image Processing Software Conversion, Vol. I & II. Development Center, Air Force Systems Command, Grifiss AFB, New York 13441. February 1977. (ADA038136 & ADA038137).

Operating Procedures

This gives the ordinary routine for running RAZMA.TAZ. Specialized uses and procedures are covered under sub-heading Miscellaneous Operations.

1. Program materials you will need include:

   User's Guide.

   IPS diskcartridge (RKØ5) (AAT Lab-blue label).

   "Scratch" diskcartiridge (RKØ5) (AAT Lab-#6).

   ROLLIN magtape with RAZMA.TAZ program and 24 picture data sets.

   IPS/DICOMED PDP-1445 computer and associated hardware (AAT lab).

2. How to 'power-up':

   a. Place PDP-11/45 console HALT switch down, turn key of console clock-wise to the 'on' position.

   b. Load IPS diskpack in RKØ5 unit #Ø. Load 'scratch' diskpack in RKØ5 unit #1. Do not use 'write protect' switch. Wait until 'on-cylinder' light comes on.

   c. Mount ROLLIN tape on tape drive. This can be omitted if program and data files are already on the scratch diskcartridge or the RPØ disk drive.

   d. Turn on RAMTEK graphics display box.

   e. Slowly turn on line printer, then toggle 'master clear', and toggle 'on-line' switches. This can be omitted if no printer output is expected (normally none is).

f. Open door to RPØ disk drive to see if diskpack #1 (DOS) is installed. If not, retrieve it from the lower cabinet and install it. Turn on power switch and wait until yellow light comes on.

g. Turn on Tektronix 4014 terminal and place it in 'local' mode. Type 'ESC' key, then ':' key. Type in that order, and only once each. This reduces the display front size. Place terminal 'on-line'.

h. Turn on CONRAC video display.

i. Return to the PDP-11/45 console SWR. Set in 000776_8 ('one' is up) and toggle 'load ADR' switch. Set in 000777_8 and toggle 'DEP' switch. Move switch from 'HALT' to 'ENABLE' and hit the 'RUN' switch.

j. Place the thumbwheel of the AED8000 to the 'MAINTENANCE' display and to ADDR CODE = 776732. Push the red IPL switch once. The LED's above the display should begin flashing. If not, go back and try again.

k. Return to the Tektronix terminal and type RKØ: Do not type a carriage return. Enter the date and time in the format shown.

l. Login with $ LO 11,11.

m. Type $ RUN DK1: RAZMA.TAZ, or $ RUN DPØ: RAZMA.TAZ.

n. Follow the yellow brick road.

3. How to 'power-down':

a. To end RAZMA.TAZ, type the 'CTRL' key and the 'C' key simultaneously, followed by 'K' and 'I'. This exits to the DOS monitor. This may also be used to stop the program at any time. Next, type 'FI' to log-out from the IPS/DOS system.

b. Place the terminal in 'LOCAL' mode, give the screen display several RESET's, and turn off the terminal.

c. Turn off the CONRAC video monitor.

d. Return to the PDP-11/45 console and hit the 'HALT' switch.

e. Toggle the switches on both RKØ5 diskcartridge drives to the UNLOAD position. The doors will remain locked until the 'LOAD' light comes on. This will take about a minute. Then the cartridges can be removed.

f. Meanwhile, the following steps may be performed.

g. Toggle the Line printer to 'OFF LINE' and turn it off.

h. Turn off the RAMTEK graphics display box.

i. Push the red IPL switch on the AED8000 once. Push, then release, the yellow power switch on the RPØ disk drive. The yellow light will slowly go out. After it goes out the diskpack can be removed if desired.

j. Unload/dismount the ROLLIN magtape, if it was loaded on the magtape drive, and turn off the magtape drive.

k. Return to the RKØ5 diskcartridge drives and remove the diskcartridges.

l. Turn the key on the PDP-11-45 console fully counterclockwise to the off position.

4. Running the RAZMA.TAZ. The actual operation of the program is straight forward. A menu of functions is presented first and the user selects the desired function. All prompts are self-explanatory. The format for all ASSIGN commands is given prior to each request for user assignment. Any error in syntax will be caught by the monitor system. However, be very careful to note the device unit number used in the command format. Failure to use the same number when typing in the command can destroy certain necessary stored program files along with the user's specified input file. Should the wrong number be typed, simply retype the command before typing 'CO'. If 'CO' has already been typed, do a 'CNTRL/C', 'KI' immediately. This closes all files and returns control to the DOS monitor. By running FILDMP or

PIP (see DOS/BATCH handbook), the user's files can be verified and/or restored. For verification of program files, see Special Procedures. After file verification or restoration, RAZMA.TAZ can be started normally.

Other than prompts and asking for file assingments there should be no other messages, except when using the FFT, POWER, and IFFT functions. In these cases, an error message will be printed about an overflow on a real to integer conversion. This is normal. It occurs during the generation of each function's image file, and refers to the DC term of the FFT coefficients. It does not affect any in ternal processing calculations. The error merely means that the value of the DC term is greater than 16,384 and exceeds the allowable 256 level gray scale of the DICOMED image processor. The DC pixel value will be zero (black pixel), in the upper left hand corner of any picture created from the image file.

Errors of any other type should not occur. If they do, it is most likely due to processing a bad data from an input file. As before, use FILDMPT or PIP to check input files for correct data.

5. RAZMA.TAZ functions. This briefly describes what each function does and what files are used or generated.

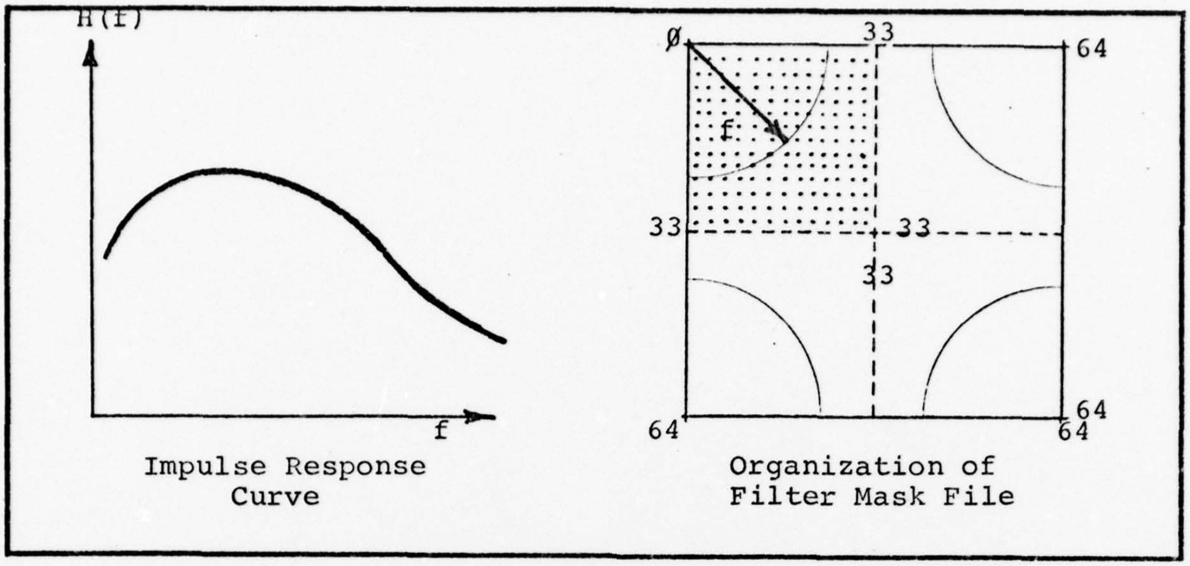
a. REDUCTION. Takes a user specified 2048x2048 image file and averages it's pixels into a 64x64 image file. The routine takes a 32x32 pixel block from the specified file and averages the entire 1024 points into one point of the 64x64 image. The output file is called AVG.IMG.

b. FFT. Performs a two-dimensional fast fourier transform (FFT) on either a user specified image file or the AVG.IMG. file. The resultant coefficients can be either normalized (FFTCOF.NRM file) or unnormalized (FFTCOF.UNN file) as requested by the user. The routine performs the FFT by using the subroutine FOURT, first on each image row, and then on each image column. An image file of the resultant FFT coefficients is also made (FFT.IMG). The DC term is in the upper left-hand corner. This image file may be accessed in the normal manner under the IPS/DICOMED system.

c. POWER. Uses the FFT coefficients to generate a two-dimensional power spectrum. The input file can be a single user specified FFT file, or FFTCOF.NRM, or FFTCOF.UNN. Additionally, the function will calculate the average power spectrum of the 24 storage picture coefficients. The power calculation is simply the squaring of the FFT coefficients. The resultant output file is PWRFFT.COF. An image file is also created (PWRFTT.IMG) for use on the IPS/DICOMED system.

d. THRESHOLD. This function looks a user specified coefficient file, or PWRFFT.COF, or FFTCOF.NRM, or FFTCOF.UMM and creates a file that when used with the MULT function will pass only coefficients above a certain threshold value. This value is entered, interactively, by the user. In addition, the routine counts the number of points above the threshold and calculates a "bandwidth" number; i.e., PTS.ABOVE ÷ 4096 = BW. Also, if the coefficients were POWER coefficients and normalized, it finds the mean-square-error of the thresholded power spectrum; i.e., the amount of power below the threshold setting. The output file contains only a mask of 1's and Ø's at the appropriate location and is called THRHLD.FIL.

e. FILTER. This function takes a user specified filter transfer function equation and computes a two-dimensional mask file which can be used in the MULT program. The value of the mask points are based on the radial distance from the upper left-hand corner of a 32x32 square (Fig 2).



Impulse Response Curve

Organization of Filter Mask File

Figure 2. User Specified Filter Impulse Response

The equation is specified on line 515 of the RAZMA.TAZ program listing. To change the equation, the program must be edited and reassembled (see Miscellaneous Operations, b). The resultant output file is FILTER.COF.

f. MULT. This functional routine takes one user specified coefficient file, or THRHOLD.COF, or FILTER. COF and multiplies it point-by-point times another user specified coefficient file, or FFTCOF.NRM, or FFTCOF.UNN, or PWRFFT.COF. It's output file is MULT.DAT.

g. IFFT. This function performs an inverse FFT on the MULT.DAT file. The user cannot directly specify input to this file. But he can however, input indirectly. By giving the THRESHOLD function a threshold of zero, a mask of all one's will be generated. Then, using this mask with MULT and specifying the other MULT input file, the input to IFFT is assured. The output is an image file named PROCSD.IMG. As with all other image files, this is operable on the IPS/DICOMED system.

6. FLOWCHART. The flowchart in Figure 3 shows the processing flow and options available in RAZMA.TAZ.

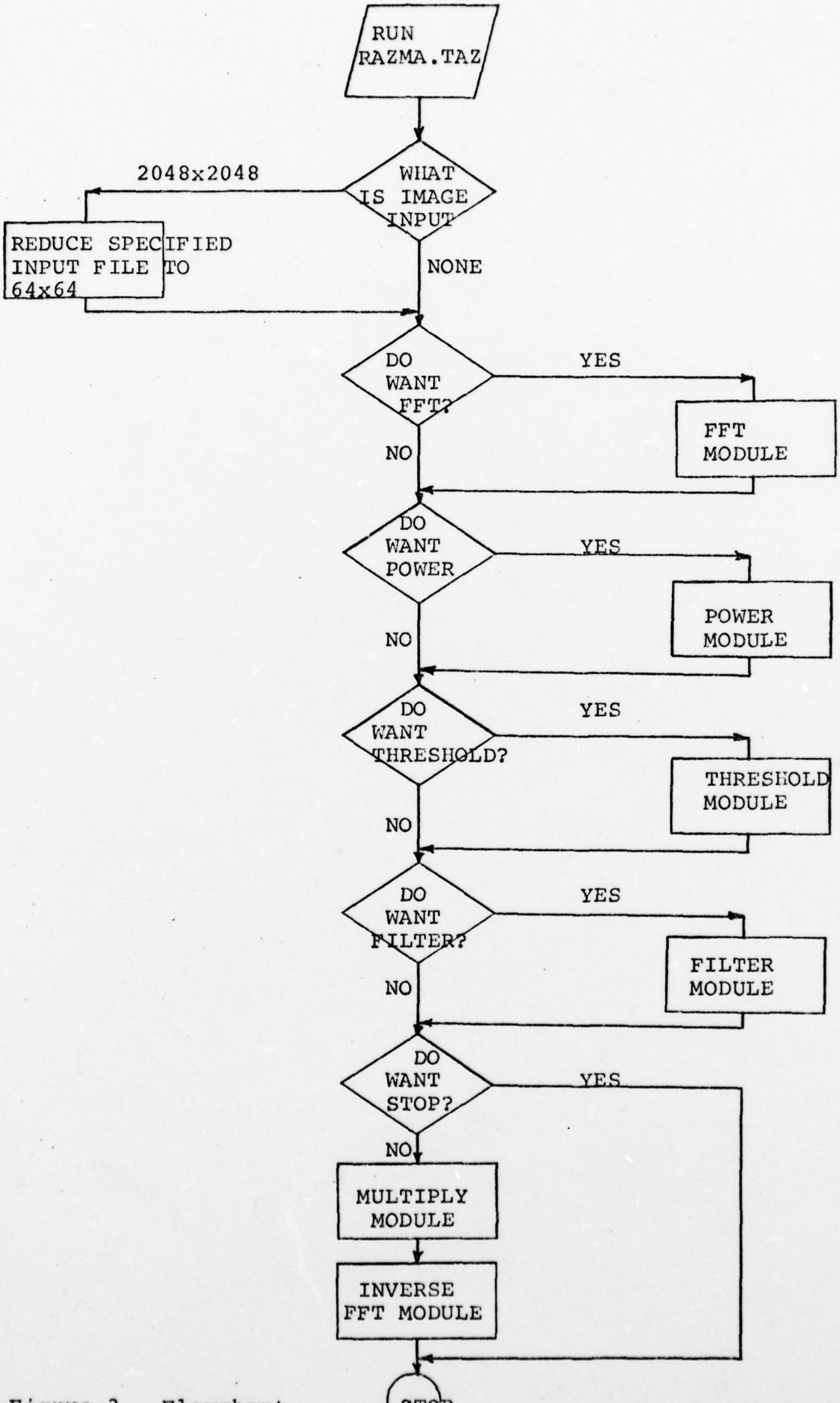
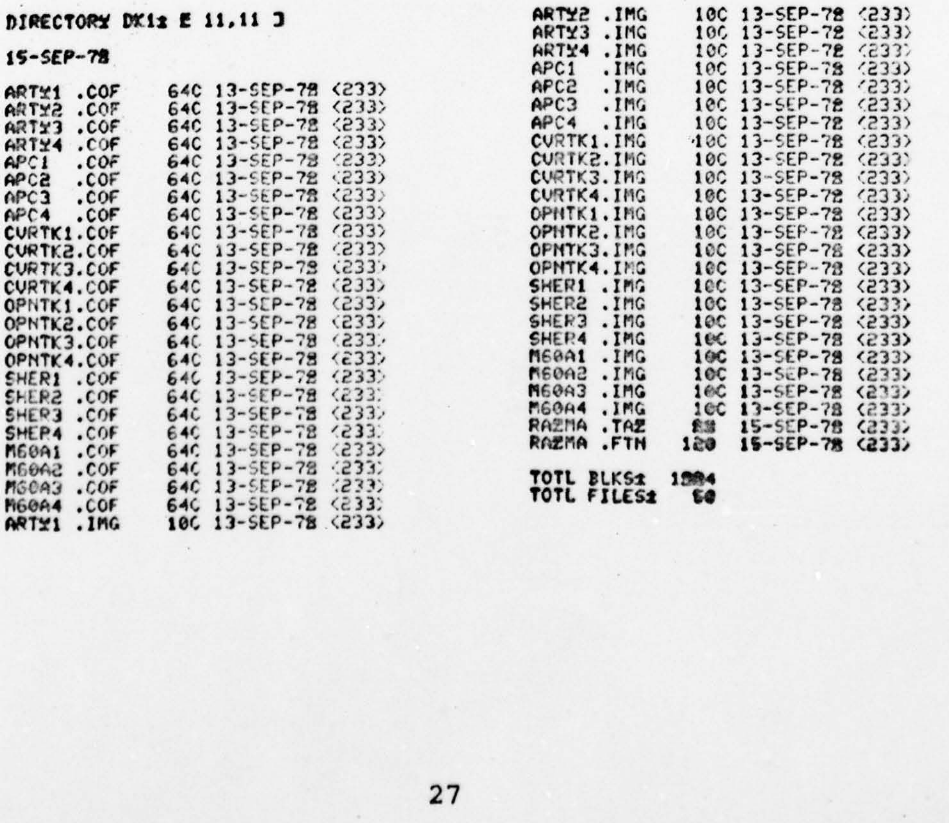


Figure 3. Flowchart

Special Procedures

1. NO FILE! After the $LO and $RUN the monitor answers with NO FILE! When this occurs, run the DOS/BATCH PIP program (see the handbook) and see if RAZMA.TAZ is on the disk you specified in the $RUN command. If not, check the other disk. Remember, you must be logged-in under UIC=11,11. If not on either disk, use the ROLLIN tape and ROLLIN program (in handbook) to restore the RKØ5 diskcartridge on unit #1 (DK1:). Then start POWER-UP procedures

2. UNABLE TO OPEN FILE. This can occur any time during the running of the RAZMA.TAZ program. This usually means that an expected input file (specified by ASSIGN) or one of the 24 standard picture files does either not exist on DPØ: or two files exist with the same name. In either case, run PIP to get a listing of all files on DPØ: to see if all required files are there, and perform file manifulation as appropriate. If file cannot be found on DPØ: or DK1:, it may be necessary to ROLLIN the magtape as in the above step and start again. A minimum of the following files are required:



```
DIRECTORY DK1: [ 11,11 ]

15-SEP-78

ARTY1 .COF      64C 13-SEP-78 <233>
ARTY2 .COF      64C 13-SEP-78 <233>
ARTY3 .COF      64C 13-SEP-78 <233>
ARTY4 .COF      64C 13-SEP-78 <233>
APC1  .COF      64C 13-SEP-78 <233>
APC2  .COF      64C 13-SEP-78 <233>
APC3  .COF      64C 13-SEP-78 <233>
APC4  .COF      64C 13-SEP-78 <233>
CVRTK1.COF      64C 13-SEP-78 <233>
CVRTK2.COF      64C 13-SEP-78 <233>
CVRTK3.COF      64C 13-SEP-78 <233>
CVRTK4.COF      64C 13-SEP-78 <233>
OPNTK1.COF      64C 13-SEP-78 <233>
OPNTK2.COF      64C 13-SEP-78 <233>
OPNTK3.COF      64C 13-SEP-78 <233>
OPNTK4.COF      64C 13-SEP-78 <233>
SHER1 .COF      64C 13-SEP-78 <233>
SHER2 .COF      64C 13-SEP-78 <233>
SHER3 .COF      64C 13-SEP-78 <233>
SHER4 .COF      64C 13-SEP-78 <233>
M60A1 .COF      64C 13-SEP-78 <233>
M60A2 .COF      64C 13-SEP-78 <233>
M60A3 .COF      64C 13-SEP-78 <233>
M60A4 .COF      64C 13-SEP-78 <233>
ARTY1 .IMG      10C 13-SEP-78 <233>
ARTY2 .IMG      10C 13-SEP-78 <233>
ARTY3 .IMG      10C 13-SEP-78 <233>
ARTY4 .IMG      10C 13-SEP-78 <233>
APC1  .IMG      10C 13-SEP-78 <233>
APC2  .IMG      10C 13-SEP-78 <233>
APC3  .IMG      10C 13-SEP-78 <233>
APC4  .IMG      10C 13-SEP-78 <233>
CVRTK1.IMG      10C 13-SEP-78 <233>
CVRTK2.IMG      10C 13-SEP-78 <233>
CVRTK3.IMG      10C 13-SEP-78 <233>
CVRTK4.IMG      10C 13-SEP-78 <233>
OPNTK1.IMG      10C 13-SEP-78 <233>
OPNTK2.IMG      10C 13-SEP-78 <233>
OPNTK3.IMG      10C 13-SEP-78 <233>
OPNTK4.IMG      10C 13-SEP-78 <233>
SHER1 .IMG      10C 13-SEP-78 <233>
SHER2 .IMG      10C 13-SEP-78 <233>
SHER3 .IMG      10C 13-SEP-78 <233>
SHER4 .IMG      10C 13-SEP-78 <233>
M60A1 .IMG      10C 13-SEP-78 <233>
M60A2 .IMG      10C 13-SEP-78 <233>
M60A3 .IMG      10C 13-SEP-78 <233>
M60A4 .IMG      10C 13-SEP-78 <233>
RAZMA .TAZ      88  15-SEP-78 <233>
RAZMA .FTN     120  15-SEP-78 <233>

TOTL BLKS:  1984
TOTL FILES:   50
```

3. EXPONENT OVERFLOW or OVERFLOW DURING REAL TO INTEGER CONVERSION. This can occur during the operation of all functions except FILTERING or THRESHOLDING. It is the result of processing with invalid data in the assigned input files, or bad data from one of the 24 stored image files. In either case, the contents of files can be inspected using the DOS/BATCH program FILDMP (see handbook). The organization and contents of typical image and coefficient files are shown in Appendix C. If bad data is suspected in any of the 24 noted image files, the best procedure is to use the DOS/BATCH program PIP to restore the files; an example of how to do this is shown in the next section.

4. UNABLE TO READ SPECIFIED FILE FORMAT or END OF FILE-END OF MEDIUM. This occurs when the RAZMA.TAZ program encounters files with improper file organization or length. Again, as above, use FILDMP or PIP to verify and transfer files.

Miscellaneous Operations

1. Generating Files. There are three catagories of files associated with the RAZMA.TAZ program.

a. Programming operating files. These are files generated by the program for it's own use. A discussion of these files is found in the comments of the program listing.

b. Twenty-four standard picture file sets. Each picture set consists of an image file and a normalized FFT coefficient file. These picture sets have already been generated and are stored for use by the program in the POWER function. A copy of these files is on the ROLLIN magtape.

c. User impact files. These are files specified by the user with the ASSIGN command. The file will be either the image or the FFT coefficient type.

The origin of the image files can be from the IPS/DICOMED system (2048x2048 or 64x64) or the RAZMA.TAZ program (64x64). Directions for creating image files under the IPS/DICOMED system can be found in Volume 1 of Reference 2. Creating a 64x64 image file under RAZMA.TAZ can be accomplished by using the reduction function, and then exiting to use PIP to rename the newly generated AVG.IMG file to whatever name the user desires. To save these files on magtape, PIP and ROLLIN are used. For each file, the PIP command is: #DK1: name.extn/ CO<DPØ: name.extn. Then, ROLLIN is used (DOS/ BATCH handbook) to copy DK1: and create a new ROLLIN tape.

The origin of the FFT coefficient files must be from running RAZMA.TAZ. The file is simply created by running the FFT function, and then exiting to use PIP to rename the newly created FFTCOF.NRM or FFTCOF.UNN file to whatever name the user desires. The file can then be saved in the same manner as the image files.

2. Restoring files. For RAZMA.TAZ to operate, all files must be on DPØ:. If they are not (a minimum listing appears in Special Procedures), or for some reasons the files become invalid, the files must be restored. A permanent copy of all files is on the ROLLIN magtape. This tape can be loaded onto DK1: (DOS/BATCH handbook), and then the appropriate file transferred under PIP to DPØ:. The proper PIP command is: #DPØ: name.extn/CO< DK1: name.extn.

3. Changing RAZMA.TAZ. Best advice: DON'T. However, for the brave and/or foolhardy, a bubble chart and program listing is given in Appendix B. Once the changes have been made under EDIT (see handbook), the proper Fortran assembly must be made. The proper command after $RUN FORTRAN is: #DK1: RAZMA, LP:< DK1: edit file name. extn / ON; or, #DPØ: RAZMA, LP:< DK1: edit file name. extn/ON. (Assuming EDIT was done on DK1:). This is important because RAZMA.TAZ uses single word integer variables. When the assembly is done, the linker command after $RUN LINK is: #DK1: RAZMA.TAZ<DK1: RAZMA/CC, DKØ: FTNLIB/L/E; or: #DPØ: RAZMA.TAZ<DK1: RAZMA/CC, DKØ: FTNLIB/L/E as appropriate.

<u>User's Guide</u>

<u>Sample Sessions</u>

EDIT

FORTRAN

PIP

ROLLIN

RAZMA.TAZ

```
RJ EDIT
EDIT-11  V07-03

#DK1±RAZMA.FTN<DK1±RAZMA.FTN
*F
*500AL
           DATA(N)=0.5*(1.0+ALOG(RADIAL*0.1+1.0))
*I.515.V
515        DATA(N)=0.5*(1.0+ALOG(RADIAL*0.1+1.0))
*3L
           DATA(N)=0.5*(1.0+ALOG(RADIAL*0.1+1.0))
520        CONTINUE
*-AA2L
515        DATA(N)=0.5*(1.0+ALOG(RADIAL*0.1+1.0))
520        CONTINUE
*B
*E≚

#^C
.KI

$
```

```
$RU FORTRN
FORTRAN  V06.13
‡DK1±RAZMA,LP±<DK1±RAZMA.FTN/ON
‡^C
.KI

$RU LINK

LINK V01-04
‡DK1±RAZMA.TAZ<DK1±RAZMA/CC,DK0±FTNLIB/L/E


SPACE USED 030420, SPACE FREE 061132
‡^C
.KI

$
```

```
$RU PIP
PIP  V10-02
#DK1±FILTER.IMG/CO<DP0±PROCSD.IMG   ← RENAMING FILE & STORING FILE

#DP0±FILTER.IMG/CO<DK1±FILTER.IMG   ← RESTORING FILE

#DK1±FILTER.IMG/DI

DK1±

[ 11,11 ]

FILTER.IMG     10C 15-SEP-78 <233>


#DP0±FILTER.IMG/DI

DP0±

[ 11,11 ]

FILTER.IMG      5C 15-SEP-78 <233>


$^C
.KI

$
```

```
ROLLIN V07

‡MT0±RAZMA<DK1±/RW/DATE±13-SEP-78   }   SAVE TWO COPIES of DISK
‡MT0±RAZMA<DK1±/DATE±13-SEP-78      }    ON MAGTAPE
‡
ROLLIN V07                         }
                                   }   RESTORING DISK FROM MAGTAPE
‡DK1±<MT0±RAZMA/FI                 }
‡
```

```
$RU DK1:RAZMA.TAZ

GREETINGS AND SOLUTIONS
SPEAK TO ME ABOUT IMAGE INPUT

TYPE: B FOR BIGGIE /2048X2048/
      J FOR JUNIOR /64X64/
      S FOR STANDARD SET OF 24
      N FOR NONE
J

THE INPUT IMAGE FILE MUST BE SPECIFIED.
IF NOT YET ASSIGNED, PLEASE DO SO NOW, THE FORM IS:
$AS DP0:NAME.EXT,7
THEN TYPE CO TO CONTINUE.
A005 000000
$AS DP0:CURTK4.IMG,7
$CO

TYPE:  N FOR NORMALIZED FFT COEFF
       U FOR UNNORMALIZED COEFF
N

FFT WORKING

FORT003022 REAL OUTSIDE RANGE ON REAL TO INTEGER CONVERSION
NAME     SEQ
MAIN.    00179

DONE FFT

TYPE C TO CONTINUE, TYPE S TO STOP
C

TYPE P FOR POWER, F FOR FILTER, T FOR THRESHOLD
P

TYPE: O, FOR ONLY ONE FFT INPUT: S, FOR STD 24 FFT COEFF
S

SQUARING FFT/S/
FORT003022 REAL OUTSIDE RANGE ON REAL TO INTEGER CONVERSION
NAME     SEQ
MAIN.    00321

DONE POWER

TYPE N TO MAKE NEW THRESHOLD FILE,TYPE O TO USE OLD FILE
N

TYPE:  V FOR VANILLA PLAIN FFT COEFF
       A FOR AVERAGED/SQUARED FFT INPUT
A

SPECIFY A REAL VALUE FOR THE THRESHOLD OF
 THE FORM = 0.0000 /UP TO 15 TRAILING DIGITS/
0.000001
```

```
THRESHOLD WORKING

THIS THRHLD.FIL USED A REAL THRESHOLD OF  0.0000009999999997
THE ABSOLUTE MSE WAS =  0.000034680684621
AND THE BANDWIDTH WAS= 2022/4096 OR  0.493652344

END THRESHOLD

TYPE C TO CONTINUE,TYPE S TO STOP
C

IF FFT COEFF WERE JUST NOW MADE, THE FILE IS
ALREADY ASSIGNED.  TYPE CO TO CONTINUE.
IF NOT YET ASSIGNED, PLEASE DO SO NOW.  THE FORM IS:
$AS DP0:NAME.EXT,3
THEN TYPE CO TO CONTINUE.
A005 000000
$CO

MULTIPLYING
HA-HA....THINK YOU CAN FOOL MOTHER NATURE EH!
DONE MULT

IFFT WORKING
DONE IFFT
$
```

```
$PU DK1:PAZMA.TAZ

GREETINGS AND SOLUTIONS
SPEAK TO ME ABOUT IMAGE INPUT

TYPE: B FOR BIGGIE (2048X2048)
      J FOR JUNIOR (64X64)
      S FOR STANDARD SET OF 24
      N FOR NONE
J

THE INPUT IMAGE FILE MUST BE SPECIFIED.
IF NOT YET ASSIGNED, PLEASE DO SO NOW. THE FORM IS:
$AS DP0:NAME.EXT,7
THEN TYPE CO TO CONTINUE.
A005 000000
$AS DP0:CUPTK4.IMG,7
$CO

TYPE:  N FOR NORMALIZED FFT COEFF
       U FOR UNNORMALIZED COEFF
N

FFT WORKING

FORT003022 REAL OUTSIDE RANGE ON REAL TO INTEGER CONVERSION
NAME     SEQ
MAIN.    00179

DONE FFT

TYPE C TO CONTINUE, TYPE S TO STOP
C

TYPE P FOR POWER, F FOR FILTER, T FOR THRESHOLD
T

TYPE N TO MAKE NEW THRESHOLD FILE,TYPE O TO USE OLD FILE
O

THE OLD THRESHOLD FILE MUST BE SPECIFIED.
IF ALREADY ASSIGNED, TYPE CO TO CONTINUE.
IF NOT YET ASSIGNED, PLEASE DO SO NOW. THE FORM IS:
$AS DP0:NAME.EXT,1
THEN TYPE CO TO CONTINUE.
A005 000000
$AS DP0:THRHLD.FIL,1
$CO

IF FFT COEFF WERE JUST NOW MADE, THE FILE IS
ALREADY ASSIGNED.  TYPE CO TO CONTINUE.
IF NOT YET ASSIGNED, PLEASE DO SO NOW.  THE FORM IS:
$AS DP0:NAME.EXT,3
THEN TYPE CO TO CONTINUE.
A005 000000
$CO
```

```
MULTIPLYING
AH-HA....THINK YOU CAN FOOL MOTHER NATURE EH!
DONE MULT

IFFT WORKING
DONE IFFT
$
```

```
$PU DK1:PAZMA.TAZ

GREETINGS AND SOLUTIONS
SPEAK TO ME ABOUT IMAGE INPUT

TYPE: B FOR BIGGIE (2048X2048)
      J FOR JUNIOR (64X64)
      S FOR STANDARD SET OF 24
      N FOR NONE
J

THE INPUT IMAGE FILE MUST BE SPECIFIED.
IF NOT YET ASSIGNED, PLEASE DO SO NOW, THE FORM IS:
$AS DP0:NAME.EXT,7
THEN TYPE CO TO CONTINUE.
A005 000000
$AS DP0:CURTN4.IMG,7
$CO

TYPE:  N FOR NORMALIZED FFT COEFF
       U FOR UNNORMALIZED COEFF
N

FFT WORKING

FORT003022 REAL OUTSIDE RANGE ON REAL TO INTEGER CONVERSION
NAME     SEQ
MAIN.    00179

DONE FFT

TYPE C TO CONTINUE, TYPE S TO STOP
C

TYPE P FOR POWER, F FOR FILTER, T FOR THRESHOLD
F

TYPE:  C TO USE CURRENT FILTER EQUATION FOR FILE
       O TO USE AN OLD FILTER FILE
C

MAKING FILTER
DONE FILTER

TYPE C TO CONTINUE, TYPE S TO STOP
C

IF FFT COEFF WERE JUST NOW MADE, THE FILE IS
ALREADY ASSIGNED.  TYPE CO TO CONTINUE.
IF NOT YET ASSIGNED, PLEASE DO SO NOW.  THE FORM IS:
$AS DP0:NAME.EXT,3
THEN TYPE CO TO CONTINUE.
A005 000000
$CO

MULTIPLYING
AH-HA....THINK YOU CAN FOOL MOTHER NATURE EH'
```

```
DONE MULT

IFFT WORKING
DONE IFFT
$
```

User's Guide

Program Bubble Chart

Program Listing

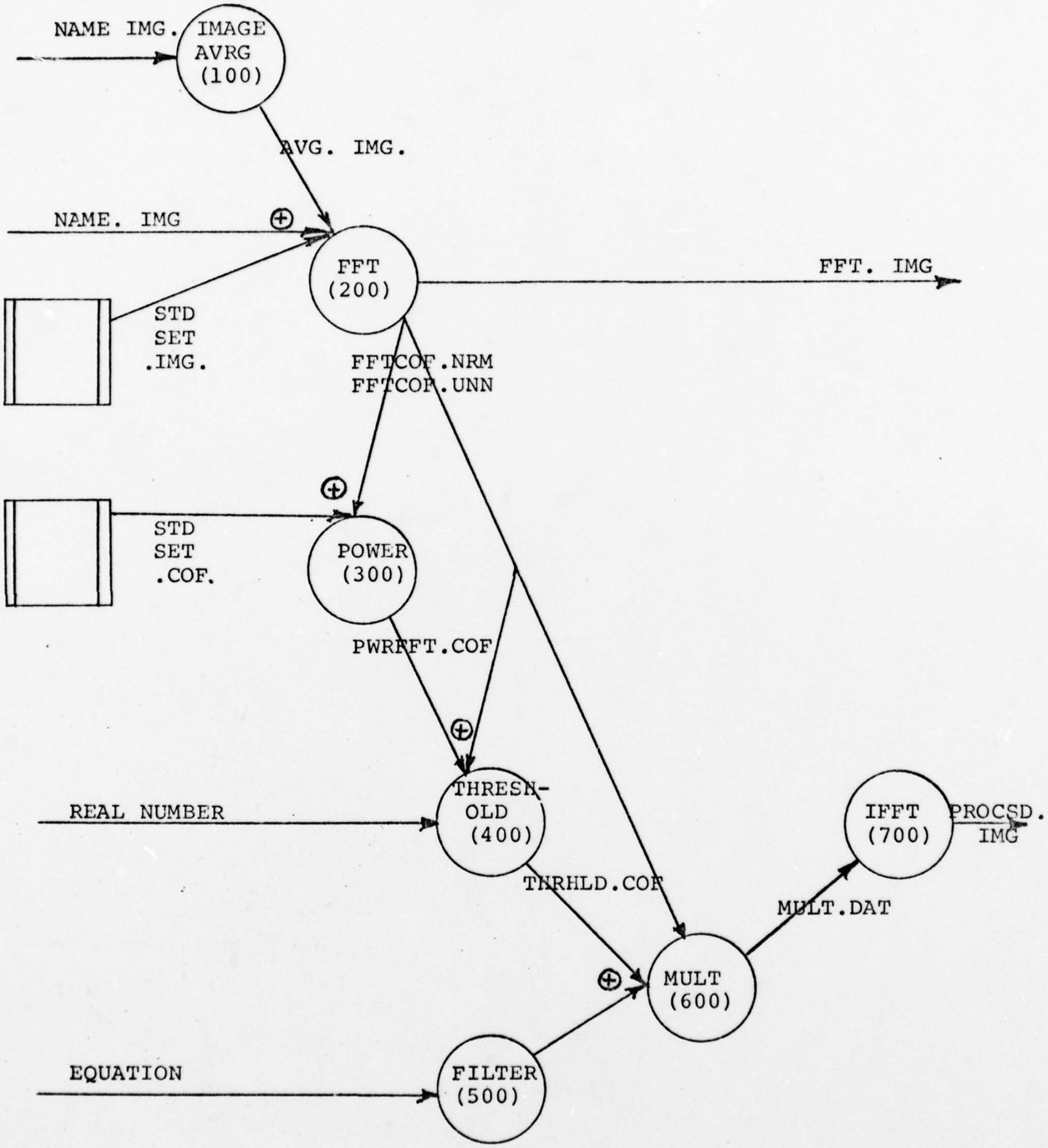


Figure 4. Program Bubble Chart

```
C       *****************************************************************
C                   PROGRAM  RAZMA.TAZ
C       *****************************************************************
C
C
C       THIS PROGRAM HAS THE FOLLOWING DISK (DPØ:) RANDOM ACCESS FILE
C       ASSIGNMENTS:

C       AVG   .IMG - OUTPUT OF REDUCE, INPUT TO FFT
C       SCRATC.DAT - USED IN FFT
C       FFTCOF.NRM - OUTPUT OF FFT, INPUT TO POWER
C       (FFTCOF.UNN)                INPUT TO THRESHOLD
C                                   INPUT TO MULT
C       FFT   .IMG - OUTPUT OF FFT
C       ------.COF -                   INPUT TO POWER
C       PWRFFT.COF - OUTPUT OF POWER, INPUT TO THRESHOLD
C       PWRFFT.IMG - OUTPUT OF POWER
C       THRHLD.FIL - OUTPUT OF THRESHOLD, INPUT TO MULT
C       FILTER.COF - OUTPUT OF FILTER, INPUT TO MULT
C       MULT  .DAT - OUTPUT OF MULT, USED IN IFFT
C       PROCSD.IMG - OUTPUT OF IFFT
C
C       THE FOLLOWING DEVICES ARE INTERACTIVELY ASSIGNED TO THESE
C       FILES DURING THE RUNNING OF THE INDICATED MODULES:
C
C       FILE          1     2     3     4     5     6     7     8
C       BIG   .IMG                                     REDUCE
C       AVG   .IMG                                      FFT
C       FFTCOF.---               MULT  THRES POWER
C       POWFFT.COF                     THRES
C       FILTER.COF  MULT
C       THRHLD.COF  MULT
C
C       ANY I/O REFERENCE MADE TO DEVICE NUMBERS AFTER THE DEVICES HAVE
C       BEEN INTERACTIVELY ASSIGNED DURING THE MODULES SPECIFIED ABOVE,
C       WILL OPERATE ON THE FILE DATA EQUATED WITH THAT DEVICE NUMBER.
C
C
C
C
C

        COMPLEX DATA(64),DATBUF(64),DCVAL,NOFILS,MSE
        BYTE HEADER(64,8),BUFFER(2048),BBUF(2),BCOL(128),NULLS(64,8)
        INTEGER INPTR,AVGPTR,DAT3PT,DAT2PT,FFTPTR,DAT7PT,DAT8PT
        INTEGER SUM,BUF,OFFSET,COL(64),UNIT,COFPTR,COFIMG,AVGFFT
        INTEGER THRSPT,OUTBW,BW,IERR,DAT1PT,DAT5PT,DAT4PT,ASGD7,PWRON
        EQUIVALENCE (BCOL(1),COL(1)),(BBUF(1),BUF)
        DATA HEADER/Ø,Ø,79,58,64,Ø,64,505*Ø/
        DATA NULLS/512*Ø/

1       WRITE (6,5)
        WRITE (6,6)
        WRITE (6,11)
2       WRITE (6,7)
        WRITE (6,8)
        WRITE (6,9)
        WRITE (6,1Ø)
        WRITE (6,11)
5       FORMAT(' GREETINGS AND SOLUTIONS ')
6       FORMAT(' SPEAK TO ME ABOUT IMAGE INPUT ')
7       FORMAT(' TYPE: B FOR BIGGIE /2Ø48X2Ø48/ ')
8       FORMAT('       J FOR JUNIOR /64X64/ ')
9       FORMAT('       S FOR STANDARD SET OF 24 ')
1Ø      FORMAT('       N FOR NONE ')
11      FORMAT('  ')
        READ (6,12) INBJSN
12      FORMAT(A)
        IF(INBJSN.EQ.1HJ) GOTO 2ØØ
        IF(INBJSN.EQ.1HB) GOTO 1ØØ
        IF(INBJSN.EQ.1HN) GOTO 3Ø
        IF(INBJSN.NE.1HS) GOTO 2
2Ø      WRITE (6,25)
        WRITE (6,26)
        WRITE (6,27)
22      WRITE (6,28)
        WRITE (6,11)
25      FORMAT(' THE STANDARD SET OF 24 PICTURES IS ALREADY STORED ')
26      FORMAT(' IN IMAGE FILES & IN NORMALIZED FFT COEFF FILES ON DPØ: ')
27      FORMAT(' IF WISH TO PROCEED TO DO POWER, FILTER, OR THRESHOLD, ')
28      FORMAT(' TYPE C TO CONTINUE, TYPE S TO STOP. ')
        READ (6,12) INCS
        IF(INCS.EQ.1HS) GOTO 9ØØ
        IF(INCS.NE.1HC) GOTO 22
3Ø      WRITE (6,35)
        WRITE (6,36)
        WRITE (6,37)
        WRITE (6,38)
        WRITE (6,11)
35      FORMAT(' TYPE: 1 TO DO FFT ')
36      FORMAT('       2 TO DO POWER ')
37      FORMAT('       3 TO DO THRESHOLDING ')
38      FORMAT('       4 TO DO FILTERING ')
        READ (6,12) IWHERE
        IF(IWHERE.EQ.1H1) GOTO 2ØØ
        IF(IWHERE.EQ.1H2) GOTO 3ØØ
        IF(IWHERE.EQ.1H3) GOTO 4ØØ
        IF(IWHERE.NE.1H4) GOTO 3Ø
        GOTO 5ØØ

C       ****** REDUCTION MODULE *****
1ØØ     WRITE (6,105)
        WRITE (6,107)
        WRITE (6,108)
        WRITE (6,1Ø9)
105     FORMAT(' THE INPUT IMAGE FILE MUST BE SPECIFIED. ')
107     FORMAT(' IF NOT YET ASSIGNED, PLEASE DO SO NOW.  THE FORM IS: ')
108     FORMAT(' $AS DPØ:NAME.EXT,7 ')
1Ø9     FORMAT(' THEN TYPE CO TO CONTINUE. ')
        ASGD7='Y'
        PAUSE
        DEFINE FILE 7(4Ø97,512,U,INPTR)
        CALL SETFIL(8,'AVG.IMG',IERR,'DP',0)
        DEFINE FILE 8(80,32,U,AVGPTR)
        WRITE (6,110)
110     FORMAT(' AVERAGE WORKING ')
        INPTR=2
        AVGPTR=1
        DO 115 J=1,8
115     WRITE (8'AVGPTR) (HEADER(I,J),I=1,64)
        DO 116 J=1,8
116     WRITE (8'AVGPTR) (NULLS(I,J),I=1,64)

        DO 170 I=1,64
          DO 120 K=1,64
120       COL(K)=0
          DO 150 J=1,32
          READ (7'INPTR) (BUFFER(N),N=1,1024)
          READ (7'INPTR) (BUFFER(N),N=1025,2048)
            DO 140 K=1,64
            OFFSET=(K-1)*32
            SUM=0
              DO 130 L=1,32
              INDX=L+OFFSET
              BBUF(1)=BUFFER(INDX)
              SUM=SUM+BUF
130           CONTINUE
            COL(K)=SUM/32+COL(K)
140          CONTINUE
150       CONTINUE
          DO 160 K=1,64
          COL(K)=COL(K)/32
160       CONTINUE
        WRITE (8'AVGPTR) (BCOL(N),N=1,128,2)
170     CONTINUE
        ENDFILE 7
        ENDFILE 8
190     WRITE (6,195)
        WRITE (6,11)
192     WRITE (6,196)
        WRITE (6,11)
195     FORMAT(' DONE IMAGE AVERAGING ')
196     FORMAT(' TYPE C TO CONTINUE, TYPE S TO STOP ')
        READ (6,12) INCS
        IF(INCS.EQ.1HS) GOTO 900
        IF(INCS.NE.1HC) GOTO 192
        GOTO 30

C       ***** FFT MODULE *****
200     CALL SETFIL(8,'AVG.IMG',IERR,'DP',0)
        DEFINE FILE 8(80,32,U,AVGPTR)
        IF(ASGD7.EQ.1HY) GOTO 215
        WRITE (6,205)
        WRITE (6,207)
        WRITE (6,208)
        WRITE (6,209)
205     FORMAT(' THE INPUT IMAGE FILE MUST BE SPECIFIED. ')
207     FORMAT(' IF NOT YET ASSIGNED, PLEASE DO SO NOW, THE FORM IS: ')
```

```
     208      FORMAT(' $AS DP0:NAME.EXT,7 ')
     209      FORMAT(' THEN TYPE CO TO CONTINUE. ')
              PAUSE
              DEFINE FILE 7(80,32,U,AVGPTR)
     210      WRITE (6,215)
              WRITE (6,216)
              WRITE (6,11)
     215      FORMAT(' TYPE:  N FOR NORMALIZED FFT COEFF ')
     216      FORMAT('        U FOR UNNORMALIZED COEFF ')
              READ (6,12) NORMUN
              DEFINE FILE 3(64,256,U,DAT3PT)
              CALL SETFIL(3,'SCRATC.DAT',IERR,'DP',0)
              DEFINE FILE 2(64,256,U,DAT2PT)
              IF(NORMUN.EQ.1HN) CALL SETFIL(2,'FFTCOF.NRM',IERR,'DP',0)
              IF(NORMUN.EQ.1HU) CALL SETFIL(2,'FFTCOF.UNN',IERR,'DP',0)
              IF(NORMUN.NE.1HU.AND.NORMUN.NE.1HN) GOTO 210
     217      DEFINE FILE 1(80,32,U,FFTPTR)
              CALL SETFIL(1,'FFT.IMG',IERR,'DP',0)
              WRITE (6,219)
     219      FORMAT(' FFT WORKING ')
              AVGPTR=17
              DAT2PT=1
              DAT3PT=1
              FFTPTR=1
              SCALE=1.0/64.0
              DO 220 J=1,8
     220      WRITE (1'FFTPTR) (HEADER(I,J),I=1,64)
              DO 225 J=1,8
     225      WRITE (1'FFTPTR) (NULLS(I,J),I=1,64)
              DO 230  K=1,64
     230      COL(K)=0

              DO 250 I=1,64
              IF(ASGD7.EQ.1HY) READ(8'AVGPTR) (BCOL(N),N=1,128,2)
              IF(ASGD7.NE.1HY) READ(7'AVGPTR) (BCOL(N),N=1,128,2)
                DO 240  J=1,64
                DATA(J)=COL(J)
     240        CONTINUE
              CALL FOURT(DATA,64,1,-1,1,0)
              WRITE (3'DAT3PT) (DATA(N),N=1,64)
     250      CONTINUE
              DO 290  K=1,64
              DAT3PT=1
                DO 260 I=1,64
                READ (3'DAT3PT) (DATBUF(N),N=1,64)
                DATA(I)=DATBUF(K)
     260        CONTINUE
              CALL FOURT(DATA,64,1,-1,1,0)
              IF(NORMUN,EQ.IHU) GOTO 270
              SCALE=256.0*64.0
              J=1
              IF(K-1) 265,265,266
     265      DCVAL=DATA(1)
              J=J+1
     266        DO 267 I=J,64
                DATA(I)=DATA(I)/DCVAL
```

```
267        CONTINUE
270      WRITE (2'DAT2PT) (DATA(N),N=1,64)
           DO 28Ø I=1,64
           COL(I)=(CABS(DATA(I)))*SCALE
28Ø        CONTINUE
         WRITE (1'FFTPTR) (BCOL(N),N=1,128,2)
29Ø      CONTINUE
         ENDFILE 1
         ENDFILE 2
         ENDFILE 3
         ENDFILE 7
         ENDFILE 8
         WRITE (6,296)
         WRITE (6,11)
295      WRITE (6,297)
         WRITE (6,11)
296      FORMAT(' DONE FFT ')
297      FORMAT(' TYPE C TO CONTINUE, TYPE S TO STOP ')
         READ (6,12) INCS
         IF(INCS.EQ.1HS) GOTO 9ØØ
         IF(INCS.NE.1HC) GOTO 295
298      WRITE (6,299)
         WRITE (6,11)
299      FORMAT(' TYPE P FOR POWER, F FOR FILTER, T FOR THRESHOLD ')
         READ (6,12) IPFT
         IF(IPFT.EQ.1HF) GOTO 5ØØ
         IF(IPFT.EQ.1HT) GOTO 4ØØ
         IF(IPFT.NE.1HP) GOTO 298
         GOTO 3ØØ

C        ***** POWER MODULE *****
3ØØ      CALL SETFIL(8,'PWRFFT.IMG',IERR,'DP',Ø)
         DEFINE FILE 8(8Ø,32,U,DAT8PT)
         DAT8PT=1
         DO 3Ø1 J=1,8
3Ø1      WRITE(8'DAT8PT) (HEADER(I,J),I=1,64)
         DO 3Ø2 J=1,8
3Ø2      WRITE(8'DAT8PT) (NULLS(I,J),I=1,64)
3Ø3      WRITE(6,3Ø4)
         WRITE (6,11)
3Ø4      FORMAT(' TYPE: O, FOR ONLY ONE FFT INPUT; S, FOR STD 24 FFT COEFF ')
         READ (6,12) INOS
         PWRON='Y'
         IF(INOS.EQ.1HS) GOTO 312
         IF(INOS.NE.1HO) GOTO 3Ø3
         IF(NORMUN.EQ.1HN) CALL SETFIL(5,'FFTCOF.NRM',IERR,'DP',Ø)
         IF(NORMUN.EQ.1HU) CALL SETFIL(5,'FFTCOF.UNN',IERR,'DP',Ø)
         IF(NORMUN.EQ.1HN.OR.NORMUN.EQ.1HU) GOTO 3Ø8
         WRITE (6,3Ø5)
         WRITE (6,3Ø6)
         WRITE (6,3Ø7)
3Ø5      FORMAT(' FFT INPUT MUST BE SPECIFIED.  PLEASE DO SO NOW. ')
3Ø6      FORMAT(' THE FORM IS:      $AS DPØ:NAME.EXT,5 ')
3Ø7      FORMAT(' THEN TYPE CO TO CONTINUE. ')
         PAUSE
3Ø8      DEFINE FILE 5(64,256,U,DAT5PT)
```

```
        CALL SETFIL(1,'PWRFFT.COF',IERR,'DP',Ø)
        DEFINE FILE 1(64,256,U,DAT1PT)
        WRITE (6,309)
 3Ø9    FORMAT(' SQUARING FFT/S/ ')
        DAT1PT=1
        DAT5PT=1
        DO 311 I=1,64
        READ (5'DAT5PT) (DATA(N),N=1,64)
          DO 31Ø J=1,64
          DATA(J)=DATA(J)*DATA(J)
          COL(J)=(CABS(DATA(J)))
 31Ø      CONTINUE
        WRITE (1'DAT1PT) (DATA(N),N=1,64)
        WRITE(8'DAT8PT) (BCOL(N),N=1,128,2)
 311    CONTINUE
        ENDFILE 1
        ENDFILE 5
        ENDFILE 8
        GOTO 39Ø

 312    WRITE (6,3Ø9)
        CALL SETFIL(5,'PWRFFT.COF',IERR,'DP',Ø)
        DEFINE FILE 5(64,256,U,DAT5PT)
        SCALE=64.Ø**3
        NOFILS=(24.Ø,24.Ø)
        DO 313 I=1,64
        DATBUF(I)=(Ø.Ø,Ø.Ø)
 313    CONTINUE
        DAT5PT=1
        DO 314 I=1,64
        WRITE (5'DAT5PT) (DATBUF(N),N=1,64)
 314    CONTINUE
        UNIT=5
        DO 36Ø  I=1,24
        DAT5PT=1
        COFPTR=1
        IF(UNIT-4) 319,315,317
 315      DO 316 J=1,4
 316      ENDFILE J
 317      DO 318 J=1,4
 318      DEFINE FILE J(64,256,U,COFPTR)
        UNIT=Ø
 319    UNIT=UNIT+1
 32Ø    IF(I-1) 322,321,322
 321    CALL SETFIL(1,'APC1.COF',IERR,'DP',Ø)
        CALL SETFIL(2,'APC2.COF',IERR,'DP',Ø)
        CALL SETFIL(3,'APC3.COF',IERR,'DP',Ø)
        CALL SETFIL(4,'APC4.COF',IERR,'DP',Ø)
 322    IF(I-5) 335,323,324
 323    CALL SETFIL(1,'ARTY1.COF',IERR,'DP',Ø)
        CALL SETFIL(2,'ARTY2.COF',IERR,'DP',Ø)
        CALL SETFIL(3,'ARTY3.COF',IERR,'DP',Ø)
        CALL SETFIL(4,'ARTY4.COF',IERR,'DP',Ø)
 324    IF(I-9) 335,325,326
 325    CALL SETFIL(1,'CVRTK1.COF',IERR,'DP',Ø)
        CALL SETFIL(2,'CVRTK2.COF',IERR,'DP',Ø)
        CALL SETFIL(3,'CVRTK3.COF',IERR,'DP',Ø)
        CALL SETFIL(4,'CVRTK4.COF',IERR,'DP',Ø)
326     IF(I-13) 335,327,328
327     CALL SETFIL(1,'OPNTK1.COF',IERR,'DP',Ø)
        CALL SETFIL(2,'OPNTK2.COF',IERR,'DP',Ø)
        CALL SETFIL(3,'OPNTK3.COF',IERR,'DP',Ø)
        CALL SETFIL(4,'OPNTK4.COF',IERR,'DP',Ø)
328     IF(I-17) 335,329,33Ø
329     CALL SETFIL(1,'M6ØA1.COF',IERR,'DP',Ø)
        CALL SETFIL(2,'M6ØA2.COF',IERR,'DP',Ø)
        CALL SETFIL(3,'M6ØA3.COF',IERR,'DP',Ø)
        CALL SETFIL(4,'M6ØA4.COF',IERR,'DP',Ø)
33Ø     IF(I-21) 335,331,335
331     CALL SETFIL(1,'SHER1.COF',IERR,'DP',Ø)
        CALL SETFIL(2,'SHER2.COF',IERR,'DP',Ø)
        CALL SETFIL(3,'SHER3.COF',IERR,'DP',Ø)
        CALL SETFIL(4,'SHER4.COF',IERR,'DP',Ø)
335     CONTINUE
          DO 35Ø J=1,64
          READ(5'DAT5PT) (DATBUF(N),N=1,64)
          READ(UNIT'COFPTR) (DATA(N),N=1,64)
            DO 337 K=1,64
337         DATA(K)=DATA(K)*DATA(K)
            DO 34Ø K=1,64
            DATBUF(K)=DATBUF(K)+DATA(K)
34Ø         CONTINUE
          DAT5PT=DAT5PT-1
          WRITE(5'DAT5PT) (DATBUF(N),N=1,64)
35Ø       CONTINUE
36Ø     CONTINUE
365     DAT5PT=1
        DO 38Ø I=1,64
        READ(5'DAT5PT) (DATA(N),N=1,64)
          DO 37Ø J=1,64
          DATA(J)=DATA(J)/NOFILS
          COL(J)=(CABS(DATA(J)))*SCALE
37Ø       CONTINUE
        DAT5PT=DAT5PT-1
        WRITE(5'DAT5PT) (DATA(N),N=1,64)
        WRITE(8'DAT8PT) (BCOL(N),N=1,128,2)
38Ø     CONTINUE
        DO 385 I=1,5
385     ENDFILE I
        ENDFILE 8
39Ø     WRITE (6,395)
        WRITE (6,11)
395     FORMAT(' DONE POWER ')
        GOTO 4ØØ

C       ***** THRESHOLD MODULE *****
4ØØ     WRITE (6,4Ø5)
        WRITE(6,11)
4Ø5     FORMAT(' TYPE N TO MAKE NEW THRESHOLD FILE,TYPE O TO USE OLD FILE ')
        READ (6,12) INNO
        IF(INNO.EQ.1HO) GOTO 6ØØ
        IF(INNO.NE.1HN) GOTO 4ØØ
```

```
    41Ø       WRITE (6,415)
              WRITE (6,416)
              WRITE (6,11)
    415       FORMAT(' TYPE:  V FOR VANILLA PLAIN FFT COEFF ')
    416       FORMAT('        A FOR AVERAGED/SQUARED FFT INPUT ')
              READ (6,12) INVA
              IF(INVA.EQ.1HA) CALL SETFIL(4,'PWRFFT.COF',IERR,'DP',Ø)
              IF(INVA.EQ.1HV.AND.NORMUN.EQ.1HN)
              1CALL SETFIL(4,'FFTCOF.NRM',IERR,'DP',Ø)
              IF(INVA.EQ.1HV.AND.NORMUN.EQ.1HU)
              1CALL SETFIL(4,'FFTCOF.UNN',IERR,'DP',Ø)
              IF(INVA.NE.1HA.AND.INVA.NE.1HV) GOTO 41Ø
              IF(NORMUN.EQ.1HN.OR.NORMUN.EQ.1HU.OR.PWRON.EQ.1HY) GOTO 43Ø
              WRITE (6,424)
              WRITE (6,425)
              WRITE (6,426)
              WRITE (6,427)
              WRITE (6,428)
    424       FORMAT(' THE FILE TO BE THRESHOLDED MUST BE SPECIFIED. ')
    425       FORMAT(' IF NOT YET ASSIGNED, PLEASE DO SO NOW. ')
    426       FORMAT(' THE FORM IS: ')
    427       FORMAT(' $AS DPØ:NAME.EXT,4 ')
    428       FORMAT(' THEN TYPE CO TO CONTINUE ')
              PAUSE
    43Ø       DEFINE FILE 4(64,256,U,DAT4PT)
              CALL SETFIL(1,'THRHLD.FIL',IERR,'DP',Ø)
              DEFINE FILE 1(64,256,U,THRSPT)
              WRITE (6,431)
    432       WRITE (6,433)
              WRITE (6,11)
    431       FORMAT(' SPECIFY A REAL VALUE FOR THE THRESHOLD OF ')
    433       FORMAT('  THE FORM = Ø.ØØØØ /UP TO 15 TRAILING DIGITS/ ')
              READ (6,435,ERR=432) THRES
    435       FORMAT(F18.15)
              WRITE(6,439)
              WRITE (6,11)
    439       FORMAT(' THRESHOLD WORKING ')

              DAT4PT=1
              THRSPT=1
              BW=Ø
              MSE=(Ø.Ø,Ø.Ø)
              DO 47Ø I=1,64
              READ (4'DAT4PT) (DATA(N),N=1,64)
                DO 46Ø J=1,64
                DATBUF(J)=(1.Ø,Ø.Ø)
                BELOW=CABS(DATA(J))-THRES
    45Ø         IF(BELOW) 451,453,453
    451         MSE=MSE+DATA(J)
                OUTBW=OUTBW+1
                DATBUF(J)=(Ø.Ø,Ø.Ø)
    453         CONTINUE
    46Ø         CONTINUE
              WRITE(1'THRSPT) (DATBUF(N),N=1,64)
    47Ø       CONTINUE
              ENDFILE 1
```

```
        ENDFILE 4
        BW=4096-OUTBW
        ABSMSE=CABS(MSE)
        FRACBW=FLOAT(BW)/4096.0

        WRITE(6,480) THRES
        WRITE (6,482) ABSMSE
        WRITE (6,484) BW,FRACBW
        WRITE (6,11)
480     FORMAT(' THIS THRHLD.FIL USED A REAL THRESHOLD OF ',F18.15)
482     FORMAT(' THE ABSOLUTE MSE WAS = ',F18.15)
484     FORMAT(' AND THE BANDWIDTH WAS= ',I4,'/4096 OR ',F12.9)
490     WRITE (6,495)
        WRITE (6,11)
491     WRITE (6,496)
        WRITE (6,11)
495     FORMAT(' END THRESHOLD ')
496     FORMAT(' TYPE C TO CONTINUE,TYPE S TO STOP ')
        READ (6,12) INCS
        IF(INCS.EQ.1HS) GOTO 900
        IF(INCS.NE.1HC) GOTO 491
        GOTO 600

C       ***** FILTER MODULE *****
500     WRITE (6,505)
        WRITE (6,506)
        WRITE (6,11)
505     FORMAT(' TYPE:  C TO USE CURRENT FILTER EQUATION FOR FILE ')
506     FORMAT('        O TO USE AN OLD FILTER FILE ')
        READ (6,12) INCO
        IF(INCO.EQ.1HO) GOTO 600
        IF(INCO.NE.1HC) GOTO 500
510     WRITE (6,512)
512     FORMAT(' MAKING FILTER ')
        DEFINE FILE 1(64,256,U,DAT1PT)
        CALL SETFIL(1,'FILTER.COF',IERR,'DP',0)
        KAPA1=0
        KAPA2=0
        SIGMA1=0.
        SIGMA2=0.
        DAT1PT=1
        IFOLD=34
        DO 530  M=1,64
        MM=M
          DO 520 N=1,64
          NN=N
          IF(M.GE.IFOLD) MM=66-M
          I=MM-1
          IF(N.GE.IFOLD) NN=66-N
          J=NN-1
          SQIJ=I**2+J**2
          RADIAL=SQRT(SQIJ)
515       DATA(N)=0.5*(1.0+ALOG(RADIAL*0.1+1.0))
520       CONTINUE
        WRITE (1'DAT1PT) (DATA(N),N=1,64)
530     CONTINUE
```

```
              ENDFILE 1
              WRITE (6,545)
              WRITE (6,11)
      540     WRITE (6,546)
              WRITE (6,11)
      545     FORMAT(' DONE FILTER ')
      546     FORMAT(' TYPE C TO CONTINUE, TYPE S TO STOP ')
              READ (6,12) INCS
              IF(INCS.EQ.1HS) GOTO 900
              IF(INCS.NE.1HC) GOTO 540
              GOTO 600


      C       ***** MATRIX MULT MODULE *****
      600     IF(INNO.EQ.1HN) CALL SETFIL(1,'THRHLD.FIL',IERR,'DP',0)
              IF(INCO.EQ.1HC) CALL SETFIL(1,'FILTER.COF',IERR,'DP',0)
              IF(INNO.NE.1HO.AND.INCO.NE.1HO) GOTO 620
              IF(INNO.EQ.1HO) WRITE (6,605)
              IF(INCO.EQ.1HO) WRITE (6,606)
      605     FORMAT(' THE OLD THRESHOLD FILE MUST BE SPECIFIED. ')
      606     FORMAT(' THE OLD FILTER FILE MUST BE SPECIFIED. ')
              WRITE (6,607)
              WRITE (6,608)
              WRITE (6,609)
              WRITE (6,610)
      607     FORMAT(' IF ALREADY ASSIGNED, TYPE CO TO CONTINUE. ')
      608     FORMAT(' IF NOT YET ASSIGNED, PLEASE DO SO NOW. THE FORM IS: ')
      609     FORMAT(' $AS DP0:NAME.EXT,1 ')
      610     FORMAT(' THEN TYPE CO TO CONTINUE. ')
              PAUSE
      620     DEFINE FILE 1(64,256,U,DAT1PT)
              IF(NORMUN.EQ.1HU) CALL SETFIL(3,'FFTCOF.UNN',IERR,'DP',0)
              IF(NORMUN.EQ.1HN) CALL SETFIL(3,'FFTCOF.NRM',IERR,'DP',0)
              WRITE (6,625)
              WRITE (6,626)
              WRITE (6,627)
              WRITE (6,628)
              WRITE (6,629)
      625     FORMAT(' IF FFT COEFF WERE JUST NOW MADE, THE FILE IS ')
      626     FORMAT(' ALREADY ASSIGNED.  TYPE CO TO CONTINUE. ')
      627     FORMAT(' IF NOT YET ASSIGNED, PLEASE DO SO NOW.  THE FORM IS: ')
      628     FORMAT(' $AS DP0:NAME.EXT,3 ')
      629     FORMAT(' THEN TYPE CO TO CONTINUE. ')
              PAUSE
              DEFINE FILE 3(64,256,U,DAT3PT)
              IF(NORMUN.EQ.1HU.OR.NORMUN.EQ.1HN) GOTO 640
      630     WRITE (6,635)
              WRITE (6,11)
      635     FORMAT(' TYPE N FOR NORM FFT COEFF, TYPE U FOR UNNORMALIZED COEFF ')
              READ (6,12) NORMUN
              IF(NORMUN.NE.1HU.AND.NORMUN.NE.1HN) GOTO 630
      640     CALL SETFIL(2,'MULT.DAT',IERR,'DP',0)
              DEFINE FILE 2(64,256,U,DAT2PT)
              WRITE (6,645)
      645     FORMAT(' MULTIPLYING ')
              DAT1PT=1
```

```
          DAT2PT=1
          DAT3PT=1
          DO 680  I=1,64
          READ (3'DAT3PT) (DATA(N),N=1,64)
          READ (1'DAT1PT) (DATBUF(N),N=1,64)
          IF(NORMUN.EQ.1HU) GOTO 665
          K=1
          IF(I-1) 650,650,655
    650   WRITE (6,651)
    651   FORMAT(' AH-HA....,THINK YOU CAN FOOL MOTHER NATURE EH! ')
          DCVAL=DATA(1)
          K=K+1
    655     DO 660 J=K,64
            DATA(J)=DATA(J)*DCVAL
    660     CONTINUE
    665     DO 670 J=1,64
            DATA(J)=DATA(J)*DATBUF(J)
    670     CONTINUE
          WRITE (2'DAT2PT) (DATA(N),N=1,64)
    680   CONTINUE
          ENDFILE 1
          ENDFILE 2
          ENDFILE 3
          WRITE (6,695)
          WRITE (6,11)
    695   FORMAT(' DONE MULT ')
          GOTO 700

    C     ***** INVERSE FFT MODULE *****
    700   CALL SETFIL(2,'MULT.DAT',IERR,'DP',0)
          DEFINE FILE 2(64,256,U,DAT2PT)
          CALL SETFIL(8,'PROCSD.IMG',IERR,'DP',0)
          DEFINE FILE 8(80,32,U,DAT8PT)
          WRITE (6,730)
    730   FORMAT (' IFFT WORKING ')
          DAT2PT=1
          DAT8PT=1
          SCALE=1.0/4096.0
          DO 740 J=1,8
          WRITE (8'DAT8PT) (HEADER(I,J),I=1,64)
    740   CONTINUE
          DO 745 J=1,8
          WRITE (8'DAT8PT) (NULLS(I,J),I=1,64)
    745   CONTINUE
          DO 750 I=1,64
          READ (2'DAT2PT) (DATA(N),N=1,64)
          CALL FOURT(DATA,64,1,1,1,0)
          DAT2PT=DAT2PT-1
          WRITE (2'DAT2PT) (DATA(N),N=1,64)
    750   CONTINUE

          DO 780  K=1,64
          DAT2PT=1
            DO 760 I=1,64
            READ (2'DAT2PT) (DATBUF(N),N=1,64)
            DATA(I)=DATBUF(K)
```

```
760         CONTINUE
          CALL FOURT(DATA,64,1,1,1,0)
            DO 770 I=1,64
            COL(I)=(REAL(DATA(I)))*SCALE
770         CONTINUE
          WRITE (8'DAT8PT) (BCOL(N),N=1,128,2)
780       CONTINUE
          ENDFILE 2
          ENDFILE 8
          WRITE (6,795)
795       FORMAT(' DONE IFFT ')
900       END

ROUTINES CALLED:
SETFIL, FOURT , CABS  , FLOAT , SQRT  , ALOG  , REAL


OPTIONS =/ON,/OP:3

BLOCK        LENGTH
MAIN.    9645   (045532)*

**COMPILER ----- CORE**
   PHASE        USED  FREE
DECLARATIVES 00963 14349
EXECUTABLES  03343 11969
ASSEMBLY     04731 15221
```

```
      SUBROUTINE FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)
C
C     THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC FORTRAN
C
C
C     TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP(ISIGN*2*PI*SQRT(-1)
C     *((J1-1)*(K1-1)/NN(1)+(J2-1)*(K2-1)/NN(2)+...))), SUMMED FOR ALL
C     J1, K1 BETWEEN 1 AND NN(1), J2, K2 BETWEEN 1 AND NN(2), ETC.
C     THERE IS NO LIMIT TO THE NUMBER OF SUBSCRIPTS.  DATA IS A
C     MULTIDIMENSIONAL COMPLEX ARRAY WHOSE REAL AND IMAGINARY
C     PARTS ARE ADJACENT IN STORAGE, SUCH AS FORTRAN IV PLACES THEM.
C     IF ALL IMAGINARY PARTS ARE ZERO (DATA ARE DISGUISED REAL), SET
C     IFORM TO ZERO TO CUT THE RUNNING TIME BY UP TO FORTY PERCENT.
C     OTHERWISE, IFORM = +1.  THE LENGTHS OF ALL DIMENSIONS ARE
C     STORED IN ARRAY NN, OF LENGTH NDIM.  THEY MAY BE ANY POSITIVE
C     INTEGERS, THO THE PROGRAM RUNS FASTER ON COMPOSITE INTEGERS, AND
C     ESPECIALLY FAST ON NUMBERS RICH IN FACTORS OF TWO.  ISIGN IS +1
C     OR -1.  IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE (OR A +1
C     BY A -1) THE ORIGINAL DATA REAPPEAR, MULTIPLIED BY NTOT (=NN(1)*
C     NN(2)*...).  TRANSFORM VALUES ARE ALWAYS COMPLEX, AND ARE RETURNED
C     IN ARRAY DATA, REPLACING THE INPUT.  IN ADDITION, IF ALL
C     DIMENSIONS ARE NOT POWERS OF TWO, ARRAY WORK MUST BE SUPPLIED,
C     COMPLEX OF LENGTH EQUAL TO THE LARGEST NON 2**K DIMENSION.
C     OTHERWISE, REPLACE WORK BY ZERO IN THE CALLING SEQUENCE.
C     NORMAL FORTRAN DATA ORDERING IS EXPECTED, FIRST SUBSCRIPT VARYING
C     FASTEST.  ALL SUBSCRIPTS BEGIN AT ONE.
C
C     RUNNING TIME IS MUCH SHORTER THAN THE NAIVE NTOT**2, BEING
C     GIVEN BY THE FOLLOWING FORMULA.  DECOMPOSE NTOT INTO
C     2**K2 * 3**K3 * 5**K5 * ....  LET SUM2 = 2*K2, SUMF = 3*K3 + 5*K5
C     + ... AND NF = K3 + K5 + ....  THE TIME TAKEN BY A MULTI-
C     DIMENSIONAL TRANSFORM ON THESE NTOT DATA IS T = T0 + NTOT*(T1+

C     T2*SUM2+T3*SUMF+T4*NF).  ON THE CDC 3300 (FLOATING POINT ADD TIME
C     OF SIX MICROSECONDS), T = 3000 + NTOT*(500+43*SUM2+68*SUMF+
C     320*NF) MICROSECONDS ON COMPLEX DATA.  IN ADDITION, THE
C     ACCURACY IS GREATLY IMPROVED, AS THE RMS RELATIVE ERROR IS
C     BOUNDED BY 3*2**(-B)*SUM(FACTOR(J)**1.5), WHERE B IS THE NUMBER
C     OF BITS IN THE FLOATING POINT FRACTION AND FACTOR(J) ARE THE
C     PRIME FACTORS OF NTOT.
C
C     PROGRAM BY NORMAN BRENNER FROM THE BASIC PROGRAM BY CHARLES
C     RADER.  RALPH ALTER SUGGESTED THE IDEA FOR THE DIGIT REVERSAL.
C     MIT LINCOLN LABORATORY, AUGUST 1967.  THIS IS THE FASTEST AND MOST
C     VERSATILE VERSION OF THE FFT KNOWN TO THE AUTHOR.  SHORTER PRO-
C     GRAMS FOUR1 AND FOUR2 RESTRICT DIMENSION LENGTHS TO POWERS OF TWO.
C     SEE-- IEEE AUDIO TRANSACTIONS (JUNE 1967), SPECIAL ISSUE ON FFT.
C
C     THE DISCRETE FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPON THE
C     DATA.
C     1.  THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VALUES
C     MUST BE THE SAME.
C     2.  BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST REPRESENT
C     EQUISPACED POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND
C     FREQUENCY.  CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUST BE
```

```
C      TRUE THAT DELTAF=2*PI/(NN(I)*DELTAT).  OF COURSE, DELTAT NEED NOT
C      BE THE SAME FOR EVERY DIMENSION.
C      3.  CONCEPTUALLY AT LEAST, THE INPUT DATA AND THE TRANSFORM OUTPUT
C      REPRESENT SINGLE CYCLES OF PERIODIC FUNCTIONS.
C
C      EXAMPLE 1.  THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A
C      COMPLEX ARRAY DIMENSIONED 32 BY 25 BY 13 IN FORTRAN IV.
C      DIMENSION DATA(32,25,13),WORK(5Ø),NN(3)
C      COMPLEX DATA

C      DATA NN/32,25,13/
C      DO 1 I=1,32
C      DO 1 J=1,25
C      DO 1 K=1,13
C   1  DATA(I,J,K)=COMPLEX VALUE
C      CALL FOURT(DATA,NN,3,-1,1,WORK)
C
C      EXAMPLE 2.  ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRAY OF
C      LENGTH 64 IN FORTRAN II.
C      DIMENSION DATA(2,64)
C      DO 2 I=1,64
C      DATA(1,I)=REAL PART
C   2  DATA(2,I)=Ø.
C      CALL FOURT(DATA,64,1,-1,Ø,Ø)
C
       DIMENSION DATA(1),NN(1),IFACT(32),WORK(1)
C      CDC 66ØØ INITIALIZATION
       WR=Ø.
       WI=Ø.
       WSTPR=Ø.
       WSTPI=Ø.
       TWOPI=6.283185307
       IF(NDIM-1)92Ø,1,1
  1    NTOT=2
       DO 2 IDIM=1,NDIM
       IF(NN(IDIM))92Ø,92Ø,2
  2    NTOT=NTOT*NN(IDIM)
C
C      MAIN LOOP FOR EACH DIMENSION
C
       NP1=2
       DO 91Ø IDIM=1,NDIM
       N=NN(IDIM)
       NP2=NP1*N
       IF(N-1)92Ø,9ØØ,5
C
C      FACTOR N
C
  5    M=N
       NTWO=NP1
       IF=1
       IDIV=2
  1Ø   IQUOT=M/IDIV
       IREM=M-IDIV*IQUOT
       IF(IQUOT-IDIV)5Ø,11,11
```

```
      11      IF(IREM)2Ø,12,2Ø
      12      NTWO=NTWO+NTWO
              M=IQUOT
              GO TO 1Ø
      2Ø      IDIV=3
      3Ø      IQUOT=M/IDIV
              IREM=M-IDIV*IQUOT
              IF(IQUOT-IDIV)6Ø,31,31
      31      IF(IREM)4Ø,32,4Ø
      32      IFACT(IF)=IDIV
              IF=IF+1
              M=IQUOT
              GO TO 3Ø
      4Ø      IDIV=IDIV+2
              GO TO 3Ø
      5Ø      IF(IREM)6Ø,51,6Ø
      51      NTWO=NTWO+NTWO
              GO TO 7Ø
      6Ø      IFACT(IF)=M
      C
      C       SEPARATE FOUR CASES--
      C          1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH,ETC.
      C             DIMENSIONS.
      C          2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION.  METHOD--
      C             TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY CON-
      C             JUGATE SYMMETRY.
      C          3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD.  METHOD--
      C             TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE OTHER
      C             HALF BY CONJUGATE SYMMETRY.
      C          4. REAL TRANSFORM FOR THE 1ST DIMENSION, N EVEN.  METHOD--
      C             TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHOSE REAL PARTS

      C             ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY PARTS
      C             ARE THE ODD NUMBERED REAL VALUES.  SEPARATE AND SUPPLY
      C             THE SECOND HALF BY CONJUGATE SYMMETRY.
      C
      7Ø      NON2=NP1*(NP2/NTWO)
              ICASE=1
              IF(IDIM-4)71,9Ø,9Ø
      71      IF(IFORM)72,72,9Ø
      72      ICASE=2
              IF(IDIM-1)73,73,9Ø
      73      ICASE=3
              IF(NTWO-NP1)9Ø,9Ø,74
      74      ICASE=4
              NTWO=NTWO/2
              N=N/2
              NP2=NP2/2
              NTOT=NTOT/2
              I=3
              DO 8Ø J=2,NTOT
              DATA(J)=DATA(I)
      8Ø      I=I+2
      9Ø      I1RNG=NP1
              IF(ICASE-2)1ØØ,95,1ØØ
      95      I1RNG=NPØ*(1+NPREV/2)
```

```
C
C       SHUFFLE ON THE FACTORS OF TWO IN N.  AS THE SHUFFLING
C       CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED
C
  100   IF(NTWO-NP1)600,600,110
  110   NP2HF=NP2/2
        J=1

        DO 150 I2=1,NP2,NON2
        IF(J-I2)120,130,130
  120   I1MAX=I2+NON2-2
        DO 125 I1=I2,I1MAX,2
        DO 125 I3=I1,NTOT,NP2
        J3=J+I3-I2
        TEMPR=DATA(I3)
        TEMPI=DATA(I3+1)
        DATA(I3)=DATA(J3)
        DATA(I3+1)=DATA(J3+1)
        DATA(J3)=TEMPR
  125   DATA(J3+1)=TEMPI
  130   M=NP2HF
  140   IF(J-M)150,150,145
  145   J=J-M
        M=M/2
        IF(M-NON2)150,140,140
  150   J=J+M
C
C       MAIN LOOP FOR FACTORS OF TWO.  PERFORM FOURIER TRANSFORMS OF
C       LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED.  THE TWIDDLE FACTOR
C       W=EXP(ISIGN*2*PI*SQRT(-1)*M/(4*MMAX)).  CHECK FOR W=ISIGN*SQRT(-1)
C       AND REPEAT FOR W=ISIGN*SQRT(-1)*CONJUGATE(W).
C
        NON2T=NON2+NON2
        IPAR=NTWO/NP1
  310   IF(IPAR-2)350,330,320
  320   IPAR=IPAR/4
        GO TO 310
  330   DO 340 I1=1,I1RNG,2
        DO 340 J3=I1,NON2,NP1

        DO 340 K1=J3,NTOT,NON2T
        K2=K1+NON2
        TEMPR=DATA(K2)
        TEMPI=DATA(K2+1)
        DATA(K2)=DATA(K1)-TEMPR
        DATA(K2+1)=DATA(K1+1)-TEMPI
        DATA(K1)=DATA(K1)+TEMPR
  340   DATA(K1+1)=DATA(K1+1)+TEMPI
  350   MMAX=NON2
  360   IF(MMAX-NP2HF)370,600,600
  370   LMAX=MAX0(NON2T,MMAX/2)
        IF(MMAX-NON2)405,405,380
  380   THETA=-TWOPI*FLOAT(NON2)/FLOAT(4*MMAX)
        IF(ISIGN)400,390,390
  390   THETA=-THETA
  400   WR=COS(THETA)
```

```
      WI=SIN(THETA)
      WSTPR=-2.*WI*WI
      WSTPI=2.*WR*WI
405   DO 570 L=NON2,LMAX,NON2T
      M=L
      IF(MMAX-NON2)420,420,410
410   W2R=WR*WR-WI*WI
      W2I=2.*WR*WI
      W3R=W2R*WR-W2I*WI
      W3I=W2R*WI+W2I*WR
420   DO 530 I1=1,I1RNG,2
      DO 530 J3=I1,NON2,NP1
      KMIN=J3+IPAR*M
      IF(MMAX-NON2)430,430,440
430   KMIN=J3

440   KDIF=IPAR*MMAX
450   KSTEP=4*KDIF
      DO 520 K1=KMIN,NTOT,KSTEP
      K2=K1+KDIF
      K3=K2+KDIF
      K4=K3+KDIF
      IF(MMAX-NON2)460,460,480
460   U1R=DATA(K1)+DATA(K2)
      U1I=DATA(K1+1)+DATA(K2+1)
      U2R=DATA(K3)+DATA(K4)
      U2I=DATA(K3+1)+DATA(K4+1)
      U3R=DATA(K1)-DATA(K2)
      U3I=DATA(K1+1)-DATA(K2+1)
      IF(ISIGN)470,475,475
470   U4R=DATA(K3+1)-DATA(K4+1)
      U4I=DATA(K4)-DATA(K3)
      GO TO 510
475   U4R=DATA(K4+1)-DATA(K3+1)
      U4I=DATA(K3)-DATA(K4)
      GO TO 510
480   T2R=W2R*DATA(K2)-W2I*DATA(K2+1)
      T2I=W2R*DATA(K2+1)+W2I*DATA(K2)
      T3R=WR*DATA(K3)-WI*DATA(K3+1)
      T3I=WR*DATA(K3+1)+WI*DATA(K3)
      T4R=W3R*DATA(K4)-W3I*DATA(K4+1)
      T4I=W3R*DATA(K4+1)+W3I*DATA(K4)
      U1R=DATA(K1)+T2R
      U1I=DATA(K1+1)+T2I
      U2R=T3R+T4R
      U2I=T3I+T4I
      U3R=DATA(K1)-T2R

      U3I=DATA(K1+1)-T2I
      IF(ISIGN)490,500,500
490   U4R=T3I-T4I
      U4I=T4R-T3R
      GO TO 510
500   U4R=T4I-T3I
      U4I=T3R-T4R
510   DATA(K1)=U1R+U2R
```

```
          DATA(K1+1)=U1I+U2I
          DATA(K2)=U3R+U4R
          DATA(K2+1)=U3I+U4I
          DATA(K3)=U1R-U2R
          DATA(K3+1)=U1I-U2I
          DATA(K4)=U3R-U4R
    520   DATA(K4+1)=U3I-U4I
          KMIN=4*(KMIN-J3)+J3
          KDIF=KSTEP
          IF(KDIF-NP2)450,530,530
    530   CONTINUE
          M=MMAX-M
          IF(ISIGN)540,550,550
    540   TEMPR=WR
          WR=-WI
          WI=-TEMPR
          GO TO 560
    550   TEMPR=WR
          WR=WI
          WI=TEMPR
    560   IF(M-LMAX)565,565,410
    565   TEMPR=WR
          WR=WR*WSTPR-WI*WSTPI+WR

    570   WI=WI*WSTPR+TEMPR*WSTPI+WI
          IPAR=3-IPAR
          MMAX=MMAX+MMAX
          GO TO 360
    C
    C     MAIN LOOP FOR FACTORS NOT EQUAL TO TWO.  APPLY THE TWIDDLE FACTOR
    C     W=EXP(ISIGN*2*PI*SQRT(-1)*(J2-1)*(J1-J2)/(NP2*IFP1)), THEN
    C     PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE OF
    C     CONJUGATE SYMMETRIES.
    C
    600   IF(NTWO-NP2)605,700,700
    605   IFP1=NON2
          IF=1
          NP1HF=NP1/2
    610   IFP2=IFP1/IFACT(IF)
          J1RNG=NP2
          IF(ICASE-3)612,611,612
    611   J1RNG=(NP2+IFP1)/2
          J2STP=NP2/IFACT(IF)
          J1RG2=(J2STP+IFP2)/2
    612   J2MIN=1+IFP2
          IF(IFP1-NP2)615,640,640
    615   DO 635 J2=J2MIN,IFP1,IFP2
          THETA=-TWOPI*FLOAT(J2-1)/FLOAT(NP2)
          IF(ISIGN)625,620,620
    620   THETA=-THETA
    625   SINTH=SIN(THETA/2.)
          WSTPR=-2.*SINTH*SINTH
          WSTPI=SIN(THETA)
          WR=WSTPR+1.
          WI=WSTPI
```

```
        J1MIN=J2+IFP1
        DO 635 J1=J1MIN,J1RNG,IFP1
        I1MAX=J1+I1RNG-2
        DO 63Ø I1=J1,I1MAX,2
        DO 63Ø I3=I1,NTOT,NP2
        J3MAX=I3+IFP2-NP1
        DO 63Ø J3=I3,J3MAX,NP1
        TEMPR=DATA(J3)
        DATA(J3)=DATA(J3)*WR-DATA(J3+1)*WI
  63Ø   DATA(J3+1)=TEMPR*WI+DATA(J3+1)*WR
        TEMPR=WR
        WR=WR*WSTPR-WI*WSTPI+WR
  635   WI=TEMPR*WSTPI+WI*WSTPR+WI
  64Ø   THETA=-TWOPI/FLOAT(IFACT(IF))
        IF(ISIGN)65Ø,645,645
  645   THETA=-THETA
  65Ø   SINTH=SIN(THETA/2.)
        WSTPR=-2.*SINTH*SINTH
        WSTPI=SIN(THETA)
        KSTEP=2*N/IFACT(IF)
        KRANG=KSTEP*(IFACT(IF)/2)+1
        DO 698 I1=1,I1RNG,2
        DO 698 I3=I1,NTOT,NP2
        DO 69Ø KMIN=1,KRANG,KSTEP
        J1MAX=I3+J1RNG-IFP1
        DO 68Ø J1=I3,J1MAX,IFP1
        J3MAX=J1+IFP2-NP1
        DO 68Ø J3=J1,J3MAX,NP1
        J2MAX=J3+IFP1-IFP2
        K=KMIN+(J3-J1+(J1-I3)/IFACT(IF))/NP1HF
        IF(KMIN-1)655,655,665

  655   SUMR=Ø.
        SUMI=Ø.
        DO 66Ø J2=J3,J2MAX,IFP2
        SUMR=SUMR+DATA(J2)
  66Ø   SUMI=SUMI+DATA(J2+1)
        WORK(K)=SUMR
        WORK(K+1)=SUMI
        GO TO 68Ø
  665   KCONJ=K+2*(N-KMIN+1)
        J2=J2MAX
        SUMR=DATA(J2)
        SUMI=DATA(J2+1)
        OLDSR=Ø.
        OLDSI=Ø.
        J2=J2-IFP2
  67Ø   TEMPR=SUMR
        TEMPI=SUMI
        SUMR=TWOWR*SUMR-OLDSR+DATA(J2)
        SUMI=TWOWR*SUMI-OLDSI+DATA(J2+1)
        OLDSR=TEMPR
        OLDSI=TEMPI
        J2=J2-IFP2
        IF(J2-J3)675,675,67Ø
  675   TEMPR=WR*SUMR-OLDSR+DATA(J2)
```

```
      TEMPI=WI*SUMI
      WORK(K)=TEMPR-TEMPI
      WORK(KCONJ)=TEMPR+TEMPI
      TEMPR=WR*SUMI-OLDSI+DATA(J2+1)
      TEMPI=WI*SUMR
      WORK(K+1)=TEMPR+TEMPI
      WORK(KCONJ+1)=TEMPR-TEMPI

680   CONTINUE
      IF(KMIN-1)685,685,686
685   WR=WSTPR+1.
      WI=WSTPI
      GO TO 690
686   TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI+WR
      WI=TEMPR*WSTPI+WI*WSTPR+WI
690   TWOWR=WR+WR
      IF(ICASE-3)692,691,692
691   IF(IFP1-NP2)695,692,692
692   K=1
      I2MAX=I3+NP2-NP1
      DO 693 I2=I3,I2MAX,NP1
      DATA(I2)=WORK(K)
      DATA(I2+1)=WORK(K+1)
693   K=K+2
      GO TO 698
C
C     COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CON-
C     JUGATE SYMMETRIES AT EACH STAGE.
C
695   J3MAX=I3+IFP2-NP1
      DO 697 J3=I3,J3MAX,NP1
      J2MAX=J3+NP2-J2STP
      DO 697 J2=J3,J2MAX,J2STP
      J1MAX=J2+J1RG2-IFP2
      J1CNJ=J3+J2MAX+J2STP-J2
      DO 697 J1=J2,J1MAX,IFP2
      K=1+J1-I3
      DATA(J1)=WORK(K)

      DATA(J1+1)=WORK(K+1)
      IF(J1-J2)697,697,696
696   DATA(J1CNJ)=WORK(K)
      DATA(J1CNJ+1)=-WORK(K+1)
697   J1CNJ=J1CNJ-IFP2
698   CONTINUE
      IF=IF+1
      IFP1=IFP2
      IF(IFP1-NP1)700,700,610
C
C     COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY CON-
C     JUGATE SYMMETRIES.
C
700   GO TO (900,800,900,701),ICASE
701   NHALF=N
      N=N+N
```

```
         THETA=-TWOPI/FLOAT(N)
         IF(ISIGN)7Ø3,7Ø2,7Ø2
   7Ø2   THETA=-THETA
   7Ø3   SINTH=SIN(THETA/2.)
         WSTPR=-2.*SINTH*SINTH
         WSTPI=SIN(THETA)
         WR=WSTPR+1.
         WI=WSTPI
         IMIN=3
         JMIN=2*NHALF-1
         GO TO 725
   71Ø   J=JMIN
         DO 72Ø I=IMIN,NTOT,NP2
         SUMR=(DATA(I)+DATA(J))/2.
         SUMI=(DATA(I+1)+DATA(J+1))/2.

         DIFR=(DATA(I)-DATA(J))/2.
         DIFI=(DATA(I+1)-DATA(J+1))/2.
         TEMPR=WR*SUMI+WI*DIFR
         TEMPI=WI*SUMI-WR*DIFR
         DATA(I)=SUMR+TEMPR
         DATA(I+1)=DIFI+TEMPI
         DATA(J)=SUMR-TEMPR
         DATA(J+1)=-DIFI+TEMPI
   72Ø   J=J+NP2
         IMIN=IMIN+2
         JMIN=JMIN-2
         TEMPR=WR
         WR=WR*WSTPR-WI*WSTPI+WR
         WI=TEMPR*WSTPI+WI*WSTPR+WI
   725   IF(IMIN-JMIN)71Ø,73Ø,74Ø
   73Ø   IF(ISIGN)731,74Ø,74Ø
   731   DO 735 I=IMIN,NTOT,NP2
   735   DATA(I+1)=-DATA(I+1)
   74Ø   NP2=NP2+NP2
         NTOT=NTOT+NTOT
         J=NTOT+1
         IMAX=NTOT/2+1
   745   IMIN=IMAX-2*NHALF
         I=IMIN
         GO TO 755
   75Ø   DATA(J)=DATA(I)
         DATA(J+1)=-DATA(I+1)
   755   I=I+2
         J=J-2
         IF(I-IMAX)75Ø,76Ø,76Ø
   76Ø   DATA(J)=DATA(IMIN)-DATA(IMIN+1)

         DATA(J+1)=Ø.
         IF(I-J)77Ø,78Ø,78Ø
   765   DATA(J)=DATA(I)
         DATA(J+1)=DATA(I+1)
   77Ø   I=I-2
         J=J-2
         IF(I-IMIN)775,775,765
   775   DATA(J)=DATA(IMIN)+DATA(IMIN+1)
```

```
              DATA(J+1)=0.
              IMAX=IMIN
              GO TO 745
      780     DATA(1)=DATA(1)+DATA(2)
              DATA(2)=0.
              GO TO 900
      C
      C       COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY
      C       CONJUGATE SYMMETRIES.
      C
      800     IF(I1RNG-NP1)805,900,900
      805     DO 860 I3=1,NTOT,NP2
              I2MAX=I3+NP2-NP1
              DO 860 I2=I3,I2MAX,NP1
              IMIN=I2+I1RNG
              IMAX=I2+NP1-2
              JMAX=2*I3+NP1-IMIN
              IF(I2-I3)820,820,810
      810     JMAX=JMAX+NP2
      820     IF(IDIM-2)850,850,830
      830     J=JMAX+NP0
              DO 840 I=IMIN,IMAX,2
              DATA(I)=DATA(J)

              DATA(I+1)=-DATA(J+1)
      840     J=J-2
      850     J=JMAX
              DO 860 I=IMIN,IMAX,NP0
              DATA(I)=DATA(J)
              DATA(I+1)=-DATA(J+1)
      860     J=J-NP0
      C
      C       END OF LOOP ON EACH DIMENSION
      C
      900     NP0=NP1
              NP1=NP2
      910     NPREV=N
      920     RETURN
              END

      ROUTINES CALLED:
      MAX0  , FLOAT , COS   , SIN

      OPTIONS =/ON,/OP:3

      BLOCK        LENGTH
      FOURT   3472   (015440)*

      **COMPILER ----- CORE**
         PHASE        USED  FREE
      DECLARATIVES 00622 14690
      EXECUTABLES  01583 13729
      ASSEMBLY     02435 17517
```

User's Guide

File Structures

```
$RU FILDMP
FILDMP  V007B

$KB:/BY<DP0:PROCSD.IMG


  PROCSD.IMG  (BYTES)

0000000 000 000 117 072 100 000 100 000 000 000 000 000 000 000 000 002
0000020 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000040 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000060 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000100 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000120 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000140 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000160 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000200 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000220 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000240 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000260 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000300 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000320 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000340 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000360 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000400 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000420 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000440 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000460 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000500 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000520 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000540 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000560 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000600 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000620 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000640 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000660 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000700 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000720 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000740 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0000760 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001020 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001040 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001060 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001100 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001120 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001140 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001160 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001200 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001220 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001240 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001260 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
```

```
0001300 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001320 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001340 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001360 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001400 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001420 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001440 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001460 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
```

```
0001500 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001520 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001540 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001560 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001600 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001620 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001640 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001660 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001700 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001720 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001740 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0001760 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
0002000 244 237 240 240 241 240 236 237 241 242 242 244 245 244 244 244
0002020 243 237 242 244 244 245 245 242 242 240 233 236 240 227 234 231
0002040 244 241 237 242 233 220 221 213 214 212 217 221 207 173 205 212
0002060 177 200 160 164 164 226 206 162 160 152 150 161 140 152 153 134
0002100 063 055 057 070 035 045 121 060 065 015 027 031 007 035 006 006
0002120 007 140 031 005 002 377 042 033 373 001 002 002 377 372 375 374
0002140 010 023 001 031 104 070 012 075 371 006 034 050 021 001 026 025
0002160 023 000 375 023 035 067 063 002 032 021 047 043 057 060 051 062
0002200 217 044 016 014 007 007 031 030 014 022 037 017 022 072 021 013
0002220 046 075 074 060 041 025 027 036 026 013 016 047 072 017 020 015
0002240 017 021 011 016 024 014 016 026 021 042 067 070 034 033 052 054
0002260 057 043 030 026 041 024 044 022 076 030 052 031 072 073 064 072
0002300 111 013 146 071 015 025 022 013 001 110 046 034 030 027 031 106
0002320 051 052 035 041 054 047 033 045 063 016 027 031 031 034 034 020
0002340 021 021 015 021 030 036 023 026 023 040 067 064 065 044 062 064
0002360 053 055 055 071 153 144 067 221 167 057 066 034 063 034 033 033
0002400 034 024 051 033 026 020 076 245 223 023 012 022 033 050 047 142
0002420 061 054 022 041 075 042 040 033 055 042 035 035 032 031 034 033
0002440 027 026 041 016 025 030 026 024 023 064 203 060 071 071 046 063
0002460 063 055 037 026 053 076 144 100 023 041 040 053 050 036 044 056
0002500 032 015 022 030 022 016 224 242 253 127 020 036 034 037 024 052
0002520 073 064 024 023 107 065 067 032 054 044 107 040 034 036 032 026
0002540 036 043 077 053 031 031 026 046 034 142 171 071 066 070 017 027
0002560 051 055 034 073 035 036 074 033 025 045 042 043 044 044 040 053
0002600 055 015 047 036 026 105 246 236 247 147 015 037 041 045 032 056
0002620 076 060 032 026 132 102 074 064 073 076 012 041 053 033 033 040
```

```
$PU FILDMP

FILDMP  V007B

$KB±.OC<DP0±FFTCOF.NRM


  FFTCOF.NRM  (OCTAL)

0000000 045002  175440  000000  000000  037312  155514  037400  173003
0000020 037022  157344  037222  031311  036741  174323  037235  052222
0000040 036215  057562  037042  132372  036501  014214  037010  157414
0000060 036630  121376  036702  050307  036712  061206  036754  171657
0000100 036611  033273  036636  153774  036634  170571  036763  164023
0000120 036274  026312  037012  035440  036446  150767  036616  104337
0000140 036123  101264  036643  115152  036552  121160  036617  034403
0000160 036153  113553  036431  170146  035564  077230  036610  063530
0000200 036135  073641  036417  060456  036112  101130  036304  001052
0000220 036306  157026  036425  146343  036160  044021  036377  166471
0000240 036265  145576  036235  137524  036307  037360  036074  056205
0000260 036354  122622  036205  133104  036234  115151  135742  036133
0000300 036035  010503  035430  150275  036374  127114  035653  127673
0000320 036236  140550  135567  107735  036276  100436  136066  067576
0000340 036404  171241  135475  127377  036325  062321  035645  113150
0000360 036432  040742  135274  144764  036315  110711  035424  012445
0000400 036446  040317  000000  000000  036315  110711  135424  012445
0000420 036432  040742  035274  144764  036325  062321  135645  113150
0000440 036404  171241  035475  127377  036276  100436  036066  067576
0000460 036236  140550  035567  107735  036374  127114  135653  127673
0000500 036035  010503  135430  150275  036234  115151  035742  036133
0000520 036354  122622  136205  133104  036307  037360  136074  056205
0000540 036265  145576  136235  137524  036160  044021  136377  166471
0000560 036306  157026  136425  146343  036112  101130  136304  001052
0000600 036135  073641  136417  060456  035564  077230  136610  063530
0000620 036153  113553  136431  170146  036552  121160  136617  034403
0000640 036123  101264  136643  115152  036446  150767  136616  104337
0000660 036274  026312  137012  035440  036634  170571  136763  164023
0000700 036611  033273  136636  153774  036712  061206  136754  171657
0000720 036630  121376  136702  050307  036501  014214  137010  157414
0000740 036215  057562  137042  132372  036741  174323  137235  052222
0000760 037022  157344  137222  031311  037312  155514  137400  173003
0001000 037344  107017  137013  122755  036735  056537  136620  [illegible]
0001020 036413  161634  036426  046340  136672  043455  036730  [illegible]
0001040 136407  124644  136420  115701  035751  065074  135277  [illegible]
0001060 036240  074776  135754  030603  135634  020670  136273  [illegible]
0001100 036117  035120  136052  163312  136215  103047  136312  [illegible]
0001120 136225  055624  136307  127556  035376  156330  136340  [illegible]
0001140 136156  122575  136254  137013  136022  045066  135002  [illegible]
0001160 035176  050527  135623  066300  136272  172757  136205  142014
0001200 135151  067017  135633  045634  035372  030312  135744  [illegible]
0001220 [illegible]  067033  136211  116070  036105  061231  036061  [illegible]
0001240 [illegible]  163432  136017  [illegible]  135307  [illegible]  136027  [illegible]
0001260 136100  146453  035445  [illegible]  [illegible]  141345  136407  [illegible]
```

Vita

Dennis A. McGaugh was born on November 9, 1947 in Eugene, Oregon. He graduated from South Eugene High School in 1966. After spending three years at Oregon State University, he was drafted into the Army in 1969. He received basic and advanced infantry training at Ft. Ord, California and then entered Infantry Officer's Candidate School at Ft. Benning, Georgia where he graduated in November 1970, and was commissioned a Second Lieutenant in Military Intelligence. During his first assignment at NSA, Ft. Meade, Maryland, he attended the John Hopkins University, Baltimore and graduated in September 1973 with a Bachelor of Science in Mechanical Engineering. His next assignment was to USASA Field Station, Korea, were he served a the operations officer of the 146th ASA Aviation Company from October 1974 until November 1975. He then attended the US Army Military Intelligence Officer's Advanced Course at Ft. Huadruca, Arizonia before entering the Air Force Institute of Technology in September 1976.

Permanent Address: 2359 Lariat Drive
Eugene, Oregon 97405

This thesis was typed by Mrs. Margaret Voigt.

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER AFIT/GE/EE/78S-16	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) AN INTERACTIVE ANALYSIS PROGRAM FOR A DIGITAL IMAGE PROCESSING LABORATORY		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Dennis A. McGaugh Capt USA		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology(AFIT/EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE September 1978 13. NUMBER OF PAGES 66
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES Approved for public release;

1-19-79

JOSEPH P. HIPPS, Major, USAF
Director of Information

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Image Processing System DICOMED
Fourier Transformation(s) IPS
Fast Fourier Transformation(s) FFT
Linear Image Filtering DFT
Power Spectrum IFFT

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A FORTRAN program is presented that will interactively prepare digitized images (pictures) for experimental analysis. The digital preparation includes the ability to Discrete Fourier Transform (DFT), Inverse Fourier Transform (IDFT), threshold image power spectral density, and to perform two-dimensional image filtering. Further, a brief discussion of image distortion measurements, optimal image size for experimental

next page

20. presentation, and image aliasing is provided. A sample of typical processed image results is also supplied.